7/9/14 **(Item 1 from file: 15)**
DIALOG(R)File 15:ABI/Inform(R)

02375387 126532331

Issues in transaction-time temporal object database systems

Norvag, Kjetil

Journal of Database Management v12n4 PP: 40-51 Oct-Dec 2001 CODEN: JDAMEQ ISSN: 1063-8016 JRNL CODE: DAN
DOC TYPE: Periodical; Feature LANGUAGE: English RECORD TYPE: Fulltext
LENGTH: 12 Pages
SPECIAL FEATURE: Illustration Graph Formula
WORD COUNT: 8155

ABSTRACT: Object database systems (ODBs) are an attractive alternative to relational database systems, especially in application areas where the modeling power or performance of relational database systems is insufficient. These applications typically maintain large amounts of data. Frequently, some of the data is temporal data. For the temporal data, the whole history of the individual objects is kept, and data is never deleted. The area of temporal ODBs is still immature, and there are many design issues that need to be solved in order to be able to achieve the desired performance. In this paper, some temporal ODB research issues and possible solutions related to object storage, object management, main memory buffering, and language bindings are discussed.

TEXT: Object database systems (ODBs) are an attractive alternative to relational database systems, especially in application areas where the modeling power or performance of relational database systems is insufficient. These applications typically maintain large amounts of data. Frequently, some of the data is temporal data. For the temporal data, the whole history ofthe individual objects is kept, and data is never deleted. The area of temporal ODBs is still immature, and there are many design issues that need to be solved in order to be able to achieve the desired performance. In this paper, we discuss some temporal ODB research issues and possible solutions related to object storage, object management, main memory buffering, and language bindings.

Object database systems (ODBs) are an attractive alternative to relational database systems, especially in application areas where the modeling power or performance of relational database systems is insufficient. These applications typically maintain large amounts of data, and additionally, often want to manage temporal data. For the temporal data, the whole history of the individual objects is kept, and data is never deleted.

Application areas of temporal object database systems include GIS systems (geographical information systems), scientific and statistical databases, multimedia systems, PACS (picture archiving and communications systems), and XML warehouses (for example Xyleme (Aguilera et al., 2000)).

The area of temporal ODBs is still immature, and most of the work has been done on data models and query languages. However, designing and implementing a system is something completely different, and introduces new problems that have to be solved. In this paper, we discuss problems and possible solutions derived from the Vagabond project at the Norwegian [illegible] Science and Technology. We will concentrate on issues [illegible] covered by previous work (discussed in the related works section), and in particular consider issues which are particular for temporal object database systems, compared to temporal relational database systems.

Transaction-Time Temporal Object Database Systems in a Nutshell

In a transaction-time temporal object database system (ODB), every object is associated with time, and an object can exist in several versions, each version being valid in a certain time interval. Every update creates a new object version. The new version is called the current version, while the previous versions are called historical versions. This versioning, related to time, is supported and maintained by the system The system also provides support for querying the temporal data.

In a non-temporal ODB, space is allocated for an object when the object is created, and updates to the objects are done in-place. This implies that after an object update, the previous version of the object is not available. In an ODB, an object is uniquely identified by a logical object identifier (OID), and an OlD index (OIDX) is used to map from the OID to the physical location of an object (Some systems use a physical OID, which means that the disk page of the object is given directly from the OID. However, in a temporal ODB, logical OIDs is the only reasonable alternative, because of objects being moved.) The entries in the OIDX, the object descriptors (ODs), contain administrative information, including information to do the mapping from logical OlD to physical address. In a non-temporal ODB, the physical location of the new version is the same as the previous version, hence, the OIDX needs only to be updated when objects are created and when they are deleted. In a temporal ODB on the other hand, we have to either 1) write the new current version to a new location, or 2) copy the previous version to a new location before we update the current version in-place. In any case, we have to update the OIDX every time we update an object, and we use one OD for each object version.

Outline of Paper

The organization of the rest of the paper is as follows. First, we give an overview of related work. In the following section we discuss object management issues, including object storage alternatives, main memory management, OID indexing, and temporal clustering and access patterns. Then, we discuss issues related to object access and queries, including programming language bindings. Finally, we conclude the paper and outline issues for further research.

RELATED WORK

Even though temporal databases have a long history, few full scale systems have been implemented. Common for most of these, is that they have only been tested on small amounts of data, which make the scalability of the systems questionable. In most of the application areas where temporal database systems are needed, scalability is an important issue, as the amount of data will be large. In the area of temporal object database systems, we are only aware of one prototype, the POST/C++ temporal object store (Suzuki and Kitagawa, 1996), based on the Texas persistent store (see below). There have also been prototypes implemented on top of non-temporal ODBs, for example by Steiner and Norrie (Steiner and Norrie, 1997), which implemented a temporal ODB on top of O .

sub 2

No-overwrite strategies have been used in shadow-- paging recovery strategies earlier, e.g., in System R (Gray et al., 1981), but with the limite strategies have been used buffer size at that time, the performance was not satisfactory. POSTGRES also earlier, e.g., in System R (Gray et a no-overwire strategy (Stonebraker, 1987), but had also its performance problems, for several reason, the most important being the buffer force strategy used.

The Vagabond approach is based on the same approach as log-structured file systems (LFS), which was introduced by Rosenblum and Ousterhout in (Rosenblum and Ousterhout, 1991). LFS has been used as the basis for two other object managers: the Texas persistent store (Singhal et al., 1992), and as a part of the Grasshopper operating system(Hulse and Dearle, 1996). Both object stores are page based, i.e., when an object has been modified, the whole page it resides on has to be written back, while the Vagabond approach is object based.

Some of the issues in this paper have also been previously presented in (Norvag, 2000b) and (Norvag, 2000c). A more in-depth study of issues related to temporal object database systems and more details about the Vagabond approach can be found in the author's doctoral thesis (Norvag, 2000e).

OBJECT MANAGEMENT ISSUES

In this section, we consider the physical management of objects on persistent storage, main memory management, clustering aspects, OID indexing, and temporal large objects.

Object Storage

A temporal database system can be implemented either through a stratum or an integrated approach. With the stratum approach, the database system is built on top of a nontemporal database system, and a layer converts temporal query language statements into conventional statements that are executed by the underlying system. Although this approach makes the introduction of temporal support into existing database systems easier, we do not see it as a long-term solution, because temporal query execution with this approach can be very costly.

Using an integrated approach, there are several way to organize the storage of objects. We will now discuss two interesting alternatives, the partitioned storage approach using in-place updating, and the log-only approach.

Partitioned Storage

Storage of data in a temporal database system is not very different from storage of data in a traditional database system. However, because current data tend to be more frequently accessed than historical data, the database is often partitioned into a current store and a history store. The two stores can utilize different storage formats, and even reside on different storage media (Ahn and Snodgrass, 1988). In this way, frequently accessed data is clustered together, stored on fast storage media, while historical versions can be stored on slower but cheaper storage media. In this way, the total storage cost is reduced, similar to the goal of general storage hierarchies.

One way to implement partitioned storage in a temporal ODB, is to store current version objects clustered together (similar to a non-temporal ODB), and write historical versions sequentially to an historical object store (for example a separate file). When an object is updated,

the previous current version is copied to the history store before the new current version is update in-place. In order to be able to access the historical versions, a separate history index can be used. This index can be as simple as a B+tree using the OID of the current version object, concatenated with time, as the index key. The leaf node entry is the OID of the current version of the object, the time interval where this version was valid, and the OD of the historical version. The location of the historical version is given through the OID in the leaf node. A variant of this approach has been used in the POST/C++ temporal object store (Suzuki and Kitagawa, 1996).

Figure 1:

Log-Only Storage

In most current database systems, data is updated inplace. In order to support recovery and increase performance, write ahead logging is used. This logging defers the in-place update, but sooner or later, the update has to be done. This often results in the writing of lots of small objects, creating a write bottleneck. To avoid this, another approach is to eliminate the database completely, and use a log-only approach, based on the same philosophy as log-structured file systems, which was introduced by Rosenblum and Ousterhout (Rosenblum and Ousterhout, 1991). The log is written contiguously to the disk, in a no-overwrite way, in large blocks. This is done by writing many objects and index entries, possibly from many transactions, in one write operation. This gives good write performance, but possibly at the expense of read performance. Figure 1illustrates the

most important differences between a traditional ODB, and a log-only ODB.

Using the log-only approach also gives new opportunities to improve performance. In order to reduce storage space and disk bandwidth, objects can be compressed before they are written. With the log-only approach, objects are written to a new location every time, so that we only use as much disk space as the size of the current version written. In a system employing in-place updating, it is difficult to benefit from object compression, because the compression ratio will be different from version to version, and it is difficult to know how much space to reserve. Another important advantage with the log-only approach is fast crash recovery. Only one pass through the log is necessary. This is very important in order to achieve high availability.

Logically, the log in a log-only system is an infinite length resource, but the physical disk size is, of course, not infinite. This problem is solved by dividing the disk into large, equal sized, physical segments. When one segment is full, writing is continued in the next available segment. As data is vacuumed, deleted or migrated to tertiary storage, old segments can be reused. Dead data, in a temporal ODB most often old index nodes, will leave behind partially filled segments, the data in these near empty segments can be collected and moved to a new segment. This process, which is called cleaning, makes the old segments available for reuse. By combining cleaning with reclustering, we can get well clustered segments. In a traditional system using in-place updating, keeping old versions of objects, which is required in a transaction time temporal database system, usually means that the previous version has to be copied to a new place before update. This doubles the write cost. With the log-only approach, this is not necessary. Keeping old versions comes for free, except for the extra disk space.

In a non-temporal ODB with in-place updating of objects, the OIDX needs only to be updated when objects are created, not when they are updated. In a log-only ODB, however, the OIDX needs to be updated on every object update. This might seem bad, and can indeed make it difficult to realize an efficient non-temporal ODB based on this technique. However, in the case of a temporal ODB, the OIDX needs to be updated on every object update also if using inplace updating, because either 1) the previous or 2) the new version mustbe written to a new place. Thus, when supporting temporal data management, the indexing cost is the same in these two approaches.
Figure 2:

Previous log-only object database systems have been page server based. While this works well in many contexts, it is not ideal. By operating on page granularity, you get many of the disadvantages of traditional pager servers. For example, if clustering is bad, and only a small part of a page has been updated, it is still necessary to write back the whole page. With bad clustering, main memory buffer utilization will be bad as well. A page based log-only ODB also makes transaction management difficult. To avoid page level locking, you essentially need to have 1) a separate log anyway, or 2) use adhoc techniques to solve the problem Both solutions are likely to hurt performance and increase complexity, and have convinced us that an object based log-only ODB is the way to go. One of the objections against operating on object granularity, has been that the read cost will be prohibitively high. We have recently shown that this is not necessarily true for a log-only temporal ODB, and that with the workload we expect to be typical for temporal ODBs, the log-only temporal ODB is highly competitive with the traditional strategies (Nervig, 2000a). This is the strategy that is used in the Vagabond approach. It is important to note that one of the main reasons why previous approaches to log-only systems have not been able to achieve significant speedup compared to traditional systems, is that they have not tried to benefit from the "free object versioning" feature. It should also be noted that some of the problems in previous no-overwrite database systems have been solved in the Vagabond approach. For example, algorithms for steal/no-force buffer management, fuzzy checkpointing and fast commit have been developed.

Main Memory Management

In order to reduce disk I/O, the most recently used index pages are kept in an index page buffer. OIDX pages will in general have low locality, and in order to increase the probability of finding a certain OD needed for a mapping from OID to physical address, it is also possible to keep the most recently used index entries (the ODs) in a separate OD cache, as is done in the Shore ODB (McAuliffe, 1997). With low locality on index pages, a separate OD cache utilizes memory better, space is not wasted on large pages where only small parts of them will be used. In non-temporal ODBs, an OD cache is only useful as a read buffer, because the OIDX update cost is low anyway (it is updated append-only). In a temporal ODB, on the other hand, individual entries are to be inserted into the OIDX. In order to make it possible to do the installation of ODs into the OIDX asynchronously, we also store these new ODs in the OD cache. At commit time, the new ODs are written to the log, and are inserted lazily into the OIDX later. By doing it this way, we increase the probability of having more than one entry for each index page that is updated. This significantly reduces the OIDX update cost.

In this paper, we describe the main memory buffers to be used in Vagabond (Nervig, 1999), a temporal ODB currently under development at the Norwegian University of Science and Technology. Special emphasis is given to the OD cache, small object buffer, and the OIDX page buffers. In order to make it possible for the system to adaptively change the size ofthe buffers, buffer models will be used to decide the buffer sizes.

We will in the following sections describe the most important main memory buffers in Vagabond, as illustrated in Figure 2: 1) OD cache, 2) small object buffer, and 3) large granularity buffers (for subobjects, OIDX-, and subobject index pages).

Object Descriptor Cache

As described previously, an OD cache is used to reduce the OIDX access costs, and reduces lookup costs as well as index update costs. ODs retrieved from lookups in the OIDX are inserted into the OD cache when retrieved, and new ODs resulting from object updates are inserted into the OD cache. However, new ODs from new objects are not initially inserted into the OD cache, they are written directly to the OIDX.

Designing an efficient OD cache is not straightforward. The requirements and functionality of the OD cache require a careful design. The entries in the OD cache are of a fine granularity, which means that additional overhead data can have a larger impact on performance than it would have in a page buffer, where the additional overhead usually is very small compared to the buffered items themselves. We will now describe operations the OD cache has to support, study some aspects of the writeback of ODs to the OIDX, and then describe the architecture of the OD cache.

OD Cache Operations. The OD cache has to support the following operations:

* lookup_current (OLD) Returns the OD of the current version of the object if the OD is in the OD cache.

* lookup_most_recent (OID) Returns the most recent OD that is resident in the OD cache.

* lookup_at (OID, TIME) Returns the OD ofthe object version valid at TIME if the OD is in the OD cache.
* lookup_start (OID, TIME) Returns the OD of the object version that has starttime (commit time) TIME ifthe OD is in the OD cache.

* lookup_end (OID, TIME) Returns the OD of the object version that has an end time TIME if the OD is in the OD cache. The end time is equal to the start time of the next version of the object (or delete item if this was the last version before the object was deleted).

* insert (OD) Inserts OD into the OD cache. If this is a new current

version of an object, set the end timestamp of the current version ofthe object ifresident inthe OD cache.

* remove (OD) Removes an OD from the OD cache.

In order to iterate through versions of an object, lookup_at () and subsequent lookup_start () operations can be used.

OD Cache-to-OIDX Writeback Because of the size of each item in the OD cache, it is very important that when dirty (note that dirty in this context means dirty with respect to the OIDX, i.e., new or a modified ODs that have not yet been inserted into the OIDX. Persistent copies of the ODs have already been written together withthe objects to the log.) ODs are to be written back to the OIDX, this can be done in batch. In order to reduce the number of index pages that has to be read (installation read of OmX pages) and written, dirty ODs are sorted so that ODs that belong to the same OIDX pages can be installed into the OIDX pages at the same time. In most cases, disk seek time will also be reduced by updating the OIDX from the list of sorted ODs. This is similar to general use of an elevator algorithm when writing back pages from a page buffer.

A complicating factor for the OD cache, is the potentially large number of entries that has to be sorted. This can be time consuming. To have a sorted list of ODs, two approaches can be used:

1. When all dirty ODs from the last dirty list have been written back to the OIDX, a new list is generated by creating an array with pointers to all dirty ODs. This array is sorted, based on the OlDs, and then the ODs are asynchronously written back. The advantage with this approach, is that the extra space overhead is minimal. However, this approach has two important drawbacks:

a) ODs created after the array has been created and sorted, will have to wait until the next checkpoint intervaL even if they belong to one of the OIDX pages that is retrieved and written when the array is processed.
b) The sorting of ODs can take several seconds of CPU time.

2. A dirty OD index with ordered elements, for example a binary tree, can be used. When a new OD is created, a pointer to the new OD is inserted into the index. During each checkpoint interval, the index is processed at least once. Because new entries are inserted immediately, we avoid the problem with the previous approach, where only ODs created during the previous checkpoint interval were available for the OD cache-to-OIDX writeback process. The disadvantage of this approach is a higher space overhead. For example, using a binary tree, two pointers are needed for each dirty OD. Note that a general priority queue is not sufficient for this index. The reason is that at some point in time, we have to be guaranteed that all entries inserted before a certain time (in this case, before the previous checkpoint interval), have been processed (in this case, before we can finish a new checkpoint).

We expect the space overhead of the dirty entry index to be compensated by increased writeback efficiency, and choose the dirty entry index approach.

OD Cache Architecture. It is possible to store all ODs, dirty as well as clean, in the index tree, and use the index tree as the access path for OD cache lookups as well. However, if that approach was used, we would have to scan through the whole index tree during each checkpoint interval. If most of the entries are clean, many CPU cycles will on average be needed in order to find the dirty ODs.

A better approach is to use one lookup index for all ODs in the OD cache, in addition to the dirty OD index. The lookup index is optimized for accesses to the OD of the most recent version of an object. The dirty entries are also indexed by this index, which means they are represented in both the dirty entry index and the lookup index. The reason for this is that without this redundancy, we would in many cases have to search both indexes when doing a lookup for a recent OD.

OD Cache Lookup Index. To understand the design of the OD cache index, it is important to remember that each update of an object creates a new OD. For each object, there will be one OD for each version, and more than one of these ODs can be in the OD cache at the same time. This means that even though the most frequent lookup operation is to retrieve the most recent OD of an object, it must be possible to store the other ODs in the OD cache as well, and it must be possible to retrieve these in an efficient way.

Figure 3:
The index is based on a chained overflow hash table. The bucket to put an OD into, is chosen based on hashing the OID of the OD. In this way, all ODs of the same object (same OID) will be in the same bucket. The ODs of an object is inserted into a version tree, for example a binary tree, where time is used as the key. ODs with different OIDs can be hashed to the same **bucket** , and for each **OID** we have a **separate version** tree. The **version** trees are chained in a linear list. With an appropriate size of the hash table, the number of OIDs hashed to the same bucket should be low.

The architecture of the OD **cache** lookup index is illustrated in Figure 3. Each entry in the hash table is a pointer to a list with pointers to the version trees. As can be observed, it would be possible to include the pointer to the next binary tree in the root of each version tree. In this way, we would avoid one pointer dereference. However, this is not done, because it could make some tree operations more complicated.

When choosing an appropriate version tree, the most important goals to achieve are 1) low insert cost, especially of a new current version OD, and 2) low lookup cost for the current version OD, which will be the most frequent operation. An ordinary binary tree is one possible solution. However, a problem with storing the ODs in a binary tree, is that if entries to be inserted into the tree have monotonically increasing key values, the result will be a linked list. Unfortunately, this is exactly the case when the inserts into the OD cache is ODs of new versions: The key value TIME is constantly increasing. One solution to this problem is to use a balanced tree, for example a splay-tree or a 2-3-tree. However, this increases the insert and space cost (it is possible to implement the splay tree with the same space cost as a binary tree, but this increases the access cost), and it is not certain that this approach will reduce the average access cost. Based on the knowledge of insert pattern and average number of versions, other heuristics can perform better, for example:

* When a new current version OD is inserted, its node is made the new root of the tree, and the current version of the tree is made the left subtree of this node. Non-current ODs are inserted into the tree following the binary tree insert algorithm. With this approach, search for the current version has a low cost.

* Another option is to keep a counter c which is increased for every insert ofa new OD into the tree, and decreased for every delete from the tree (but always non-negative, i.e., if c is zero and we have a deletion, c will remain zero). If c reaches a certain threshold, the tree is reorganized and c is set to zero. Although a reorganizing approach in general is a bad idea, with higher cost than using a balanced tree, it can perform well in the OD cache if we assume that only a very few of the version trees have an insert rate that is high enough to result in reorganizations. The space overhead is low for this approach, as we only need an additional counter for each version tree.

* On average, it is even possible that a list could perform well. The problem with this approach, is the high worst case cost (Rastogi et al. used a linear list for versions of data items in the Dali main memory storage manager (Rastogi et). However, in Dali, versioning is only used to support transient versioning, and not to provide support for temporal data. Hence, the length of a version list will usually be short.).

OD Cache Replacement. The OD cache will only have empty slots during startup, before enough objects have been accessed to fill up the OD cache.

After the cache has filled up, one of the ODs resident in the OD cache has to be discarded before a new OD can be inserted. Only non-dirty ODs (with respect to the OIDX) can be discarded, and the clock algorithm is used as an LRU approximation to decide which of the candidate ODs should be discarded. The number of dirty ODs in the OD cache should be kept relatively low, to reduce the cost when searching for a candidate OD for replacement.

Small Object Buffer

In Vagabond, small objects are stored and retrieved as separate entities, and an object buffer is the only reasonable choice. (In a temporal ODB using in-place updating for current object versions, a dual buffer consisting of a combined object and page buffer can be used. In that case, the object buffer part can be implemented like the small object buffer described here.) Large objects have to be treated differently, because some of the subobject index pages and the subobjects of an object version might also be a part of other object versions. In this section aspects of the small object buffer are described, and in the next section buffering of large object subobject index pages and subobjects are described.

Modified Object Chain. For each active transaction, there is a modified object chain. This list contains the objects that have not yet been written to disk, but must be written before an commit operation can finish.

Small Object Buffer Architecture. For the objects in the small object buffer, a clock algorithm is used as an LRU approximation. The resident small object table (RSOT) is used to store administrative information on objects currently resident in the small object buffer. The access to the RSOT is through an index structure similar to the one used for lookups in the OD cache.

Although the information stored in the RSOT alternatively could be stored together with the ODs in the OD cache, the number of objects resident in memory is in general much smaller than the number of entries in the OD cache, making that approach less space efficient.

When an object is read into the buffer, its OD is removed from the OD cache, and reinserted into the OD cache when the object is discarded from the object buffer. Although this at first glance might seemto be inefficient, it simplifies the OD cache management considerably, and also has the benefit of removing interaction and synchronization between the OD cache and the small object buffer.

When a small object is retrieved, the memory location and the size of the object is inserted into the RSOT, together with the memory location and the size of the object. The reason for storing the object size in the RSOT entry, is that the object size in the OD is the size of the object while on disk. On disk the object might be compressed, and thus have a size different from the size when in main memory. Using the physical location field in the OD to store the main memory location of the object could be done to save space, but if that was done, we would have a problem when the object was discarded from the buffer. We would then have lost the log address, and the OD would have to be discarded as well. That would make it necessary to do a costly OIDX lookup the next time the object was to be accessed.

Note that when an object is updated, a new OD is created for the object. If both versions are to be stored in main memory, a new RSOT entry has to be created for the new version.

Large Granularity Buffers

The most frequently used subobjects and OIDX and subobject index pages, are buffered in the large granularity buffers. In Figure 2, separate buffers are used for each of the categories, but it is possible to use a common buffer if desired. The large granularity buffers are similar to traditional disk page buffers, where an item is retrieved from disk to the buffer on demand.

Temporal Clustering and Access Patterns

The performance of page server based ODBs depend heavily on good clustering of objects, i.e., a high probability that more than one object on a disk page retrieved from the disk will be used in the near future, before the page is discarded from the buffer. A good clustering reduces the number of object pages that has to be read and written, and it also results in better buffer memory utilization. In practice, when different applications with different access patterns access the ODB, it is difficult to achieve good clustering. In a study by Tsangaris and Naughton (Tsangaris and Naughton, 1992), all practical clustering algorithms resulted in an average clustering less than 0.25. Although less of an issue when log-only storage is used (where we do not rely that much on clustering), writing related objects together increases the gain from prefetching and disk read ahead.

In non-temporal ODBs, clustering considers different objects, and we only try to predict cases like "when object 01 is accessed, it is also likely that object 00 will be accessed shortly after." However, it is likely that in a temporal ODB application, a good object clustering includes historical object versions as well as current object versions (after all, the reason for storing the historical object versions is thatwe want to access them later!). A good example that illustrates this is that in a traditional ODB without support for temporal objects, we would often simulate object versions by including timestamp as an user managed attribute in the objects, and store the objects in an object collection. With temporal support, the user will only see one object, but can access the different object versions. Thus, even if the object versions are historical versions, it is possible that some of these will be part of the hot set of object versions. As a result, possible access patterns in a temporal ODB also include cases like:

* If the current version of object O

sub i

is accessed, it is also likely that all the historical versions of object O

sub i

will be accessed in the near future.

Figure 4:

* If the current version of object O

sub i
. is accessed, it is also likely that the previous current version of object O

sub i

will be accessed in the near future.

* If the current version of object O

sub i

is accessed, it is also likely that all the historical versions of object O

sub i

. will be accessed in the near future.

* If the version of object O

sub i
. valid at time T

sub i

. is accessed, it is also likely that the version of obj ect O

sub i

. valid at time T

sub i

. will be accessed in the near future.

Many other access patterns exist, but this illustrates the increased complexity that is introduced in the clustering process. It also shows that partitioned storage makes clustering more difficult, there is no simple and efficient way to include historical objects in the clustering together with current objects. If log-only storage is employed, clustering of temporal objects is much easier, as it facilitates adaptive reclustering during segment cleaning.
In a traditional system, it is possible for the user or database administrator to define some clustering strategy for a database, for example by defining a clustering tree or using clustering hints. These approach can also be extended to temporal ODBs, but necessitates continuous reordering, because it is impossible to reserve space for all new versions that are created. We expect that even if these explicit clustering techniques extended with time can be used, adaptive reorganization will be even more important in future systems.

OID Indexing

In a temporal ODB, it is necessary to use logical OlDs, and an OID index (O)DX) is needed to do the mapping from logical OID to physical location when retrieving an object. We aim at supporting temporal data, while still having current version access performance close to a non-temporal database system. This is difficult with the general multiversion access methods, e.g., the TSB-tree, the R-tree, or LHAM. Even if these index structures could give better support for temporal operations, we believe efficient non-temporal operations to be crucial, as they will probably still be the most frequently used operations. The use of general multiversion access methods will increase storage space considerably and index insert considerably, and many of the alternative methods will also incur a considerable degree of redundancy. TSB-trees and R-trees have both good support for time-key search. However, when indexing ODs, we use the OID as the key, and many queries will only be for perfect match, not OlD range. One problem with LHAM is that it can have high lookup cost when the current version is to be searched for. As this will be a very frequently used operation, LHAM is not suitable for our purpose. Secondary indexes, on the other hand, will typically be realized from one of the traditional multiversion access methods mentioned above.

We have designed a new index structure, the Vagabond Temporal ODD Index (VTOIDX). Our main goals in the design of the VTOIDX was.

1. Support for temporal data, while still having index performance close to a non-temporal/one version database system Even if the use of other index structures could give better support for temporal operations, we believe efficient non-temporal operations to be crucial, as they will probably still be the most frequently used operations.

2. Efficient object-relational operation. This is achieved by the use of physical containers.

3. Easy tertiary storage migration of partitions of the index. An OIDX is space consuming, a size in the order of 20% of the size of the database itself is not unreasonable. In the case of migration of old versions of

objects to tertiary storage, it is desirable, and in practice necessary, that parts of the OIDX itself can be migrated as well to avoid the need for large amounts of disk space for the OIDX of the migrated objects.

As a result, we ended up with an index structure that is an hierarchy of multi-way tree indexes, with three levels (Figure 4):

1. Container index (CONTIDX): The CONT)DX, one for each physical database, indexes the physical containers in a database. A physical container can be used, for example, to store all objects from a class, or for all objects in a collection. This can also be used to implicitly maintain class extents, and makes set based queries efficient. The pointers in the leaf nodes points to a current version OID index, one for each container.

2. Current version ODD index (CVOIDX). The CVO)DX is an index for all ODs of the current versions of objects in one container, one OD for each object version.

Figure 5:
3. Historical version subindex (HVOIDX): The CVOIDX itself is a nested tree index: The leaf nodes of the CVOIDX only contains the current version of the ODs, the previous versions are kept in separate subindex trees, which we call historical version subindex (HVOIDX). For each leaf node in the CVOIDX, there is a separate subindex tree, with non-current versions of all ODs that resides or have resided in the actual leaf node. Note that versioning is only needed on the bottom level of the hierarchy.

By having separate indexes for each container, it is easier to achieve high space utilization, because each subindex index is append-only. It is also easy to migrate a container to tertiary storage. By separating ODs of current and historical object versions, scan over the current version objects in a container is efficient. The CVOIDX and HVOIDX combination canbe viewed as an extension ofthe Surrogate-Time (ST) index proposed by Gunadhi and Segev (Gunadhi and Segev, 1993). However, instead of one HVO>DX for each object, we use one HVOIDX for each CVOIDX node. The VTOIDX is described in more detail in (Nervig, 2000d).

Temporal Large Objects

In most ODBs, all objects smaller than a disk page is stored together with other small objects in a disk page, and objects larger than a disk page are segmented into "subobjects" and accessed through a large object index, a "subobject index." In a log-only database system like Vagabond, we can use a more flexible approach, where the threshold for when to consider the object as a large object can be much smaller than one page, for example 512 bytes, or if desired, much larger, for example 128 KB.

EXODUS Large Objects

There are several ways to manage the subobjects in a temporal ODB, but it should satisfy one important requirement: storage efficient versioning, i.e., when a large object is updated, as few subobjects as possible and as few of the subobject index nodes as possible should be rewritten. This can be achieved by the use of a subobject index that also takes care of versioning, similar to the EXODUS large storage objects (Carey et al., 1986).

The EXODUS storage system supports versioning, as illustrated in Figure 5. On top of the figure is the OIDX, which has separate ODs for each version of an object (in EXODUS, physical OIDs are used, so there is not actually an OIDX). To the left we have the initial version of an indexed large object. When parts ofthe object is updated, only the updated subobjects are stored. The index structure for the new version points to the previous version of the subobjects for those parts that have not been modified. This versioning also applies to the large object indexes themselves. If the indexes have more than one level, only modified parts ofthe index are rewritten. The result is minimal duplication of subobjects, efficient update of the subobject index, and efficient access to parts of the large

objects.

Vagabond Large Objects

A Vagabond large object is based on the EXODUS large storage objects. To make large objects flexible, we have extended the EXODUS large storage object:

* Vagabond, the subobject threshold and subobject size can be different for different object classes. This is very useful, because different object classes can have different object retrieval characteristics. Tvi cal examples are a video and a collection (for example a set, bag, or array). When playing a video, you want to retrieve one large segment of the video each time. On the other hand, when searching an index tree, you only want to retrieve single nodes, which usually have a small size. Similar for both video and index retrieval is that you only want a part of the object. For other objects, the whole object will be needed at once. One example is images. In order to be able to display the image, the whole object is needed. In that case, storing the image as one contiguous object will be advantageous.

In the internal large object index nodes, we use (NavigDesc,pointer) tuples instead of (count,pointer) tuples, where the NavigDesc (Navigational Descriptor) is a variable length structure. In this way, a Vagabond large object is a generalization of the EXODUS large storage objects. EXODUS large storage objects can be realized by using a counter as the NavigDesc, this is the default case.

More complex indexes and other special objects can be implemented as more general Vagabond large objects by using more complex NavigDescs. For example, R-trees can be realized by using rectangle descriptors in the NavigDesc data structure, a (timestamp, key) tuple can be used for temporal indexes etc.

OBJECT ACCESS AND QUERIES

In non-temporal ODBMSs, ODMG's OQL or similar query languages canbe used for ad-hoc queries. Similar to the way OQL is a superset of the part of standard SQL that deals with databases queries, it is possible to design a temporal OQL that is a superset of TSQL2. One such approach has been described by Fegaras and Elmasri (Fegaras and Elmasri, 1998). However, one of the main advantages of ODBMSs is the avoidance of the language mismatch by providing computationally complete data manipulation languages with no mismatch between language and storage. In the ODMG standard, language bindings based on C++, Java and Smalltalk are described. Such language bindings are also needed for temporal ODBMSs. It should also be noted that in order to use methods in queries, these issues have to be resolved.

A general purpose programming language is only designed for current data. Integrating support for access of historical data into a programming language introduces a lot of interesting but difficult issues, including:

* Which object interface/signature to use when accessing a historical object version. The schema might have been changed since the historical version was created, so that the current interface to the class is different from the one previously used.

* Which method implementation to use when calling methods in historical objects. One straightforward approach is to use the implementation that was current at the same time as the actual object version was current. However, this is not necessarily what we want, if the reason for a new implementation of a method was a bug in the previous version. This problem can be solved by providing the necessary information at schema change time.

* How to integrate time into the syntax of the programming language.

In the rest of this section, we will discuss the integration of access to historical data into a general-purpose programming language.

Temporal C++ Binding

In this section, we describe two approaches that extend the C++ language binding with support for access to historical data in a transaction-time ODBMS. The first approach is based on the language binding used in POST/C++ (Suzuki and Kitagawa, 1996), while the second is to our knowledge new. The concepts of these approaches can also be employed for a Java language binding.

Explicit Object Version Access

The easiest way to integrate object version access into the programming language is to provide explicit access to the versions. This is the way it is done in POST/C++ (Suzuki and Kitagawa, 1996). Given an OID, the program can be given a pointer to a historical version valid at a particular time by calling a function snapshot (OLD, time). It is also possible to create iterators that can be used to navigate the versions of an object in chronological sequence.

This approach should be easy to use and understand, but if it should be possible to call a method in a historical object version that accesses other objects, the historical version must itself do the necessary operations in order to retrieve the objects valid at the same time as when the version was created.

REFERENCES

Aguilera, V., Cluet, S., Veltri, P., Vodislav, D., and Wattez, F. (2000). Querying XML documents in Xyleme. Technical Report 182, Verso/INRIA.

Ahn, I. and Snodgrass, R. (1988). Partitioned storage for temporal databases. Information Systems, 13(4).

Carey, M., DeWitt, D., Richardson, J., and Shekita, E. (1986). Object and file management in the EXODUS extensible database system. In Proceedings of the 12th VLDB Conference.
Cattell, R., editor (1997). The Object Database Standard: ODMG-93. Release 2.0. Morgan Kaufmann. Fegaras, L.and Elmasri, R. (1998). A temporal object query language. In Proceedings of the Fifth International Workshop on Temporal Representation and Reasoning.

Gray, J., McJones, P., Blasgen, M., Lindsay, B., Lorie, R., Price, T., Putzolu, F., and Traiger, I. (1981). The recoverymanager of the System R database manager. ACM Computing Surveys, 13(2).

Gunadhi, H. and Segev, A. (1993). Efficient indexing methods for temporal relations. IEEE Transactions on Knowledge and Data Engineering, 5(3).

Hulse, D. and Dearle, A. (1996). A log-structured persistent store. In Proceedings of the 19th Australasian Computer Science Conference.

McAuliffe, M. L. (1997). Storage Management Methodsfor Object Database Systems. PhD thesis, University of WisconsinMadison.

Norvag, K. (2000a). A comparative study of log-only and inplace update based temporal object database systems. In Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM'2000).
Norvag, K. (2000b). Design issues in transaction-time temporal object database systems. In Proceedings of ADBISDASFAA '2000.

Norvag, K. (2000c). Main-memory management in temporal object database systems. In Proceedings ofADBIS-DASFAA '2000.

Norvag, K. (2000d). The Vagabond temporal OID index: An index structure for OID indexing in temporal object database systems. In Proceedings of the 2000 International Database Engineering and Applications Symposium (IDEAS).

Norvag, K. (2000e). Vagabond: The Design and Analysis of a Temporal Object Database Management System. PhD thesis, Norwegian University of Science and Technology.

Norvag, K. (1999). The Vagabond parallel temporal objectoriented database system: Versatile support for future applications. In Proceedings of Norsk Informatikkonferanse 1999.

Rastogi, R., Seshadri, S., Bohannon, P., Leinbaugh, D., Silberschatz, A., and Sudarshan, S. (1997). Logical and physical versioning in main memory databases. In Proceedings of the 23rd VLDB Conference.

Rosenblum, M. and Ousterhout, J. K. (1991). The design and implementation of a log-structured file system. In Proceedings of the Thirteenth ACM Symposium on Operating System Principles.

Singhal, V., Kåkkad, S., and Wilson, P. (1992). Texas: An efficient, portable persistent store. In Proceedings of the Fifth International Workshop on Persistent Object Systems.

Steiner, A. and Norrie, M. C. (1997). Implementing temporal databases in object-oriented systems. In Proceedings of the 5th International Conference on Database Systems for Advanced Applications (DASFAA'97).

Stonebraker, M. (1987). The design of the POSTGRES storage system. In Proceedings of the 13th VLDB Conference.

Suzuki, T. and Kitagawa, H. (1996). Development and performance analysis of a temporal persistent object store POST/ C++. In Proceedings of the 7th Australasian Database Conference.

Tsangaris, M. and Naughton, J. (1992). On the performance of object clustering techniques. In Proceedings of SIGMOD'92.

KJETIL NORVAG

Norwegian University of Science and Technology, Norway

Kjetil Norvag is an Associate Professor in the Department of Computer and Information Science at the Norwegian University of Science and Technology. He received a Dr. Ing. degree in computer science from the Norwegian University of Science and Technology in 2000, and was a postdoctoral researcher in the VERSO group at INRIA (France) in 2001. His major research interests include query and storage of XML documents, object database systems, temporal database systems, query processing, and access methods.

```
? ds; show files

Set     Items   Description
S1    3084289   (CACHE? OR  TEMPORARY()(STORAGE OR  MEMORY? )  OR  DISK? -
             OR BUFFER? OR RAM?)
S2   20273776   (WRIT??? OR WRIT???()DISK? OR TRANSFER??? OR RECORD??? OR -
             ENCOD??? OR UPDAT???)
S3    2317511   (OLD OR ORIGINAL? OR PREVIOUS OR EARL??? OR OUTDAT???)(7N)-
             (NEW??? OR RECENT OR MODERN OR FRESH)
S4     564739   (MODIF???  OR MODIFICAT??? OR AMEND??? OR AMENDM??? OR CHA-
             NG??? OR ADJUST??? OR ADJUSTM???)(3N)(COP??? OR DATA OR FILE??
              OR INFORMATION?? OR INFO? OR VERSION? )
S5   20111793   (NODE? OR TERMINAL? OR COMPUTER? OR CLIENT? OR SERVER? OR -
             WORKSTATION?? OR STATION??)
S6     144140   (PERSIST??? OR PERSEVER??? OR ENDUR??? OR LINGER??? OR REM-
             AIN??? OR STEAD??? OR PROLONG??? OR UNCHANG???)(3N)(STOR??? OR
              STORAGE? OR MEMOR??? OR HARD()DRIVE OR DISK? OR CD()ROM?? OR
             CDROM OR ROM?? OR FLASH? OR EPROM?? OR PROM??)
S7    1896451   (MASTER? OR CONTROL??? OR MAIN OR PRIMARY? OR SUPERVIS??? -
             OR ADMINISTRA??? OR MANAG?)(5N) S5
S8         48   AU=(CHANDRASEKARAN, S?  OR CHANDRASEKARAN S?)
S9         69   AU=(BAMFORD, R? OR BAMFORD R? )
S10         4   AU=(BRIDGE, W? OR  BRIDGE W?)
S11       381   AU=(BROWER, D? OR  BROWER D?)
S12         3   AU=(MACNAUGHTON, N?  OR MACNAUGHTON N?)
S13       392   AU=(CHAN, W? OR  CHAN W?)
S14         0   AU=(SRIHARI, V? OR SRIHARI V?)
S15       897   S8 OR S9 OR S10 OR S11 OR S12 OR S13 OR S14
S16         0   S8 AND S9 AND S10 AND S11 AND S12 AND S13 AND S14
S17       166   S1(S)S2(S)S3(S)S4
S18        20   S1(3N)S2(5N)S3(5N)S4
S19        99   S6(3N)S7
S20         0   S6(3N)S7(3N)S3
S21         2   S6(3N)S7(3N)S2
S22        31   S1(S)S2(S)S3(S)S6
S23        14   S1(7N)S2(7N)S3(7N)S6
S24         7   S1(7N)S2(7N)S3(7N)S7
S25        43   S18 OR S21 OR S23 OR S24
S26        30   RD (unique items)
S27        25   S26 NOT PY>1999
S28        23   S27 NOT PD=19981124:200201124
S29        23   S28 NOT PD=200201124:20050103
File   9:Business & Industry(R) Jul/1994-2004/Dec 30
         (c) 2004  The Gale Group
File  15:ABI/Inform(R) 1971-2005/Jan 01
         (c) 2005 ProQuest Info&Learning
File  16:Gale Group PROMT(R) 1990-2004/Jan 03
         (c) 2004 The Gale Group
File  20:Dialog Global Reporter 1997-2005/Jan 03
         (c) 2005 The Dialog Corp.
File  47:Gale Group Magazine DB(TM) 1959-2004/Jan 03
         (c) 2004 The Gale group
File  75:TGG Management Contents(R) 86-2004/Dec W1
         (c) 2004 The Gale Group
File  80:TGG Aerospace/Def.Mkts(R) 1982-2004/Jan 03
         (c) 2004 The Gale Group
File  88:Gale Group Business A.R.T.S. 1976-2004/Dec 29
         (c) 2004 The Gale Group
File  98:General Sci Abs/Full-Text 1984-2004/Sep
         (c) 2004 The HW Wilson Co.
```

```
File 112:UBM Industry News 1998-2004/Jan 27
         (c) 2004 United Business Media
File 141:Readers Guide 1983-2004/Sep
         (c) 2004 The HW Wilson Co
File 148:Gale Group Trade & Industry DB 1976-2004/Jan 03
         (c)2004 The Gale Group
File 160:Gale Group PROMT(R) 1972-1989
         (c) 1999 The Gale Group
File 275:Gale Group Computer DB(TM) 1983-2004/Jan 03
         (c) 2004 The Gale Group
File 264:DIALOG Defense Newsletters 1989-2004/Dec 27
         (c) 2004 The Dialog Corp.
File 484:Periodical Abs Plustext 1986-2004/Dec W4
         (c) 2004 ProQuest
File 553:Wilson Bus. Abs. FullText 1982-2004/Sep
         (c) 2004 The HW Wilson Co
File 570:Gale Group MARS(R) 1984-2004/Jan 03
         (c) 2004 The Gale Group
File 608:KR/T Bus.News. 1992-2005/Jan 03
         (c)2005 Knight Ridder/Tribune Bus News
File 620:EIU:Viewswire 2004/Dec 30
         (c) 2004 Economist Intelligence Unit
File 613:PR Newswire 1999-2005/Jan 03
         (c) 2005 PR Newswire Association Inc
File 621:Gale Group New Prod.Annou.(R) 1985-2004/Jan 03
         (c) 2004 The Gale Group
File 623:Business Week 1985-2004/Dec 27
         (c) 2004 The McGraw-Hill Companies Inc
File 624:McGraw-Hill Publications 1985-2004/Dec 28
         (c) 2004 McGraw-Hill Co. Inc
File 634:San Jose Mercury  Jun 1985-2004/Dec 31
         (c) 2005 San Jose Mercury News
File 635:Business Dateline(R) 1985-2005/Jan 01
         (c) 2005 ProQuest Info&Learning
File 636:Gale Group Newsletter DB(TM) 1987-2004/Jan 03
         (c) 2004 The Gale Group
File 647:CMP  Computer Fulltext 1988-2004/Dec W2
         (c) 2004 CMP Media, LLC
File 696:DIALOG Telecom. Newsletters 1995-2004/Dec 30
         (c) 2004 The Dialog Corp.
File 674:Computer News Fulltext 1989-2004/Dec W2
         (c) 2004 IDG Communications
File 810:Business Wire 1986-1999/Feb 28
         (c) 1999 Business Wire
File 813:PR Newswire 1987-1999/Apr 30
         (c) 1999 PR Newswire Association Inc
File 587:Jane`s Defense&Aerospace 2004/Dec W3
         (c) 2004 Jane`s Information Group
?
```

29/3,K/1 (Item 1 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)

00825704 94-75096
Cache jump starts Bernoulli drives
Jonez, James A
Computer Technology Review v14n1 PP: 42 Jan 1994
ISSN: 0278-9647 JRNL CODE: CTN
WORD COUNT: 682

ABSTRACT: A **cache** makes a fast **disk** drive even faster. Improved performance in a Bernoulli drive means less time wasted for the user and greater productivity because the **computer** completes its transactions and returns **control** to the user much faster. Compared to read-only **cache** in **earlier** Bernoulli drives, the **new** **cache** is up to 3 times faster. Tests show that **write** operations for the new Bernoulli drive will match or beat typical performance of many hard...
...TEXT: performance in Bernoulli means less time wasted for the user and greater productivity because the **computer** completes its transactions and returns **control** to the user much faster. Compared to the read-only **cache** in **earlier** Bernoulli drives, the **new** **cache** is up to three times faster. Performance will vary depending on the system, application, and other factors. Testing by Iomega shows that for sequential **writes** on a DOS system, performance can be up to five times faster than some competing ...

29/3,K/2 (Item 2 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)

00716882 93-66103
What a difference a year makes: CD-ROM developments
Beiser, Karl
Online v17n3 PP: 109-111 May 1993
ISSN: 0146-5422 JRNL CODE: ONL
WORD COUNT: 2180

...TEXT: of high-capacity data storage subsystems are on the market right now. Many support the **writing** of **new** data as well as the reading of **old** data. Hard **disk** technology continues to advance rapidly as measured on a cost per megabyte basis. Nevertheless, **CD - ROM** **remains** the best compromise between cost, capacity,

29/3,K/3 (Item 3 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)

00592195 92-07368
LAN Fever: How to Protect Your Business from Computer Viruses
Brenner, Brad
Office Systems v9n1 PP: 23 Jan 1992
ISSN: 8750-3441 JRNL CODE: OFS
WORD COUNT: 747

...TEXT: system, finding a clean version is as easy as accessing the

optical disk.

"Although information **recorded** on **write** -once media can't be altered, it can be **updated** ," said Kalstrom. "Each time a file is recalled and edited, a **new** version of the file is created. The **previous** version **remains** somewhere on the **disk** . Once a virus is detected, the user need only trace the file back step-by...

29/3,K/4 (Item 1 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

05970441 Supplier Number: 53260039 (USE FORMAT 7 FOR FULLTEXT)
Web-based training solutions.(Software Review)(Evaluation)
Uiterwijk, Julie
InfoWorld, pNA(1)
Nov 23, 1998
Language: English Record Type: Fulltext
Article Type: Evaluation
Document Type: Magazine/Journal; Trade
Word Count: 9198

... changes to our course and redeployed it on the Web server: The students received the **old** **version** , not our **new** **changes** . We discovered that the program left the old course in the **cache** instead of **updating** it with the new version. If you intend to make changes to a course after...

29/3,K/5 (Item 2 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

05516823 Supplier Number: 48359747 (USE FORMAT 7 FOR FULLTEXT)
Storage: Quarterdeck Introduces DiskClone Corporate for Small and
Medium-Sized Businesses; The Fast and Safe Way to Upgrade Operating Systems and Copy Hard Disk Drives Over a Network
EDGE: Work-Group Computing Report, pN/A
March 16, 1998
Language: English Record Type: Fulltext
Document Type: Newsletter; Trade
Word Count: 831

... 1 gigabytes. Quarterdeck's Partition-It Extra Strength, with Select-It, is also included in **DiskClone** Corporate.

DiskClone Corporate Features:

Scripting Allows multiple installs over the network.

Automatic **Adjustment** of Partitions **Copies** images onto larger hard drives without requiring the drive to be partitioned. **DiskClone** divides the **new** drive into the same proportions as the **original** , copied drive.

Compression Compresses data prior to imaging, providing faster **transfer** rates and convenient storage.

FAT 32 Support Allows users of Windows 95 or 98 to...

29/3,K/6 (Item 3 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

05505597 Supplier Number: 48342407 (USE FORMAT 7 FOR FULLTEXT)
FlowPoint Corporation and Alcatel Microelectronics Sign Agreement for ADSL DynaMiTe Product Development.
Business Wire, p3091059
March 9, 1998
Language: English Record Type: Fulltext
Document Type: Newswire; Trade
Word Count: 1136

... or other programs on multiple PCs.
Its scrip-cloning process to ensure data integrity. e. **DiskClone** Corporate Features Scripting Allows multiple installs over the network. Automatic **Adjustment** of Partitions **Copies** images onto larger hard drives without requiring the drive to be partitioned. **DiskClone** divides the **new** drive into the same proportions as the **original** , copied drive. Compression Compresses data prior to imaging, providing faster **transfer** rates and convrovides security by password protecting images stored on removable disks or CDs. Manual...

29/3,K/7 (Item 1 from file: 47)
DIALOG(R)File 47:Gale Group Magazine DB(TM)

05433079 SUPPLIER NUMBER: 20631287 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Windows Utilities: Rx for Your PC.(Buyers Guide)
Andrews, Dean; McCracken, Harry; Spector, Lincoln
PC World, v16, n6, p110(1)
June, 1998
DOCUMENT TYPE: Buyers Guide ISSN: 0737-8939 LANGUAGE: English
RECORD TYPE: Fulltext
WORD COUNT: 7244 LINE COUNT: 00577

... 98 will include the troubleshooting System Configuration Utility; the Version Conflict Manager, which prevents saving **old** files over **new** ; and the System File Checker. Add/Remove Programs will **remain** , but **Disk** Cleanup will empty the Recycle Bin and Internet **cache** . Win 98 will also **update** Win 3.1's Dr. Watson diagnostics; the new Automatic Skip Driver Agent will watch...

29/3,K/8 (Item 2 from file: 47)
DIALOG(R)File 47:Gale Group Magazine DB(TM)

03782116 SUPPLIER NUMBER: 12351121 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Tune up your hard drive. (The Dedicated Mac) (Column)
Weigand, C.J.
Home Office Computing, v10, n7, p16(1)
July, 1992
DOCUMENT TYPE: Column ISSN: 0899-7373 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 800 LINE COUNT: 00061

... them are still there; you just can't find your way to them (your files **remain** accessible, though, to **disk** recovery utilities).

Reformatting, on the other hand, is like tearing down the walls and replacing them with **new** ones. All the **old** rooms (files) and their contents are gone.

If recurring problems loading programs or **writing** to disk inspire you to reformat your hard disk, back up any irreplaceable data. After...

29/3,K/9 (Item 3 from file: 47)
DIALOG(R)File 47:Gale Group Magazine DB(TM)

02521251 SUPPLIER NUMBER: 00535484 (USE FORMAT 7 OR 9 FOR FULL TEXT)
New Improved Lattice C.
Baker, L.; Sakowski, N.
PC Magazine, v3, n5, p138-141 3 Pages
March 20, 1984
DOCUMENT TYPE: evaluation ISSN: 0888-8507 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 1493 LINE COUNT: 00111

... new line command is sent or not. The upgraded I/O functions do not acquire **buffers** via level 2 memory allocation, a **change** from **previous** **versions** . In the **new** version, the MS-DOS performs its own **buffering** . The I/O functions open, read, **write** , Lseek, and close are carried out more quickly. Large read/write data transfer operations also...

29/3,K/10 (Item 1 from file: 141)
DIALOG(R)File 141:Readers Guide

03788177 H.W. WILSON RECORD NUMBER: BRGA98038177 (USE FORMAT 7 FOR FULLTEXT)
Windows utilities: Rx for your PC.
Andrews, Dean.; McCracken, Harry.; Spector, Lincoln.
PC World v. 16 no6 (June 1998) p. 110-13+
WORD COUNT: 7178

(USE FORMAT 7 FOR FULLTEXT)

TEXT:
... 98 will include the troubleshooting System Configuration Utility; the Version Conflict Manager, which prevents saving **old** files over **new** ; and the System File Checker. Add/Remove Programs will **remain** , but **Disk** Cleanup will empty the Recycle Bin and Internet **cache** . Win 98 will also **update** Win 3.1's Dr. Watson diagnostics; the new Automatic Skip Driver Agent will watch...

29/3,K/11 (Item 2 from file: 141)
DIALOG(R)File 141:Readers Guide

01512651 H.W. WILSON RECORD NUMBER: BRGA89012651
Put a positive lock on your data.
AUGMENTED TITLE: DataSentry
Miastkowski, Stan.
Byte (Byte) v. 14 (Feb. '89) p. 100

...ABSTRACT: types SEAL followed by the name of the file. DataSentry encrypts the file into a **new** file about half the size of the **original** . It then **writes** over the old file, safely eradicating any information that could **remain** on the **disk** . To restore the encrypted file, the user must reinsert the key. The basic DataSentry package...

29/3,K/12 (Item 1 from file: 148)
DIALOG(R)File 148:Gale Group Trade & Industry DB

07162872 SUPPLIER NUMBER: 15012834 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Panasonic packs higher capacity in new opticals. (LF-7300A, LF07304; LF-5300A WORM drive; LF-J7324A jukebox) (Product Announcement)
Morgenstern, David
MacWEEK, v8, n2, p6(1)
Jan 10, 1994
DOCUMENT TYPE: Product Announcement ISSN: 0892-8118 LANGUAGE: ENGLISH RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 430 LINE COUNT: 00032

... and 1.5-Gbyte rewriteable media, and 1.4-Gbyte, 940- and 470-Mbyte WORM (**write** -once-read-many) media.

Panasonic said its optical mechanisms use a phase- **change** technology that **writes** **data** in a single pass, unlike ordinary magneto-optical drives, which require one pass to erase **old** data and another to **record** the **new** . **Disks** **written** on phase-change drives are incompatible with MO media.

To increase capacity to 1.5...

29/3,K/13 (Item 1 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01897918 SUPPLIER NUMBER: 17856187 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Defragmenting an NT hard disk. (Executive Software Inc's Diskeeper for Windows NT 1.01 disk optimizing software) (Software Review)(Evaluation)
Binns, Don
PC User, n271, p97(1)
Nov 15, 1995
DOCUMENT TYPE: Evaluation ISSN: 0263-5720 LANGUAGE: English
RECORD TYPE: Fulltext; Abstract
WORD COUNT: 770 LINE COUNT: 00062

... error occurs during copying, the whole process is aborted and the original file is left **unchanged** and intact.

When **Diskeeper** is sure that the file copy is accurate, it **updates** the Master File Table to point to the **new** location and deletes the **original** file. This is the only point at which **Diskeeper** needs exclusive access to the file.

Diskeeper has been designed to run without operator intervention. The scheduler can be set to run...

29/3,K/14 (Item 2 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01520195 SUPPLIER NUMBER: 12237454 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Critics complain utility 'saves' changes - to the trash. (program errors in Alysis Software Corp.'s More Disk Space file management software) (Brief Article)
Norr, Henry
MacWEEK, v6, n22, p6(1)
June 8, 1992
DOCUMENT TYPE: Brief Article ISSN: 0892-8118 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT
WORD COUNT: 390 LINE COUNT: 00028

... decompresses sections of it into memory as needed, leaving the original compressed version intact on **disk** .

When the user makes and saves **changes** , the edited **version** is **written** to a temporary file, called SuperDisk!TempfileA (or B, C, and so on, as multiple...

...in an invisible folder called Temporary Items. Only when the document is closed is the **original** file replaced by a **new** **version** that incorporates the **changes** and takes over the document's proper name.

That means that if the user makes...

29/3,K/15 (Item 3 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01386029 SUPPLIER NUMBER: 09663825 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Ombudsman. (column)
Watkins, Margaret; Ries, Andy; Haugen, Rich; Field, J.H.; Morris, Richard; Cohen, J.
PC Sources, v1, n10, p429(3)
Oct, 1990
DOCUMENT TYPE: column ISSN: 1052-6579 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT
WORD COUNT: 2191 LINE COUNT: 00158

... Sources Ombudsman will act as your mediator, helping to resolve your dispute. To contact Ombudsman, **write** to PC Sources Ombudsman, c/o PC Sources, One Park Avenue, **New** York, NY 10016.

LOST IN THE MAIL

Early in January, I ordered a **disk** drive for my AT **computer** and a color video **controller** card for my XT **computer** from The Essence Group.

When the order came, the company had sent the wrong drive...

29/3,K/16 (Item 4 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01207555 SUPPLIER NUMBER: 06168844 (USE FORMAT 7 OR 9 FOR FULL TEXT)
More tips on using Lotus products. (Good Ideas) (column)
Ridington, Richard W., Jr.
Lotus, v3, n3, p97(4)

March, 1987
DOCUMENT TYPE: column ISSN: 8756-7334 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT
WORD COUNT: 2774 LINE COUNT: 00204

... such settings. If the disk is write-protected, 1-2-3 will display the message **Disk** is write-protected. After you leave the program, 1-2-3 erroneously attempts to update that file once more. When prevented by a **write** -protect tab, DOS returns the message Abort, Retry, Ignore? If you change disks before resolving the problem, DOS will **write** the **original** disk's directory over the **new** disk's directory. The best thing to do when you get that message is to reboot the **computer** (**Control** -Alt-Delete on most **computers**). If the new **disk** has a **write** -protect tab in place and DOS still **writes** the **original** **disk** 's directory on the **new** **disk** , either the **write** -protect tab does not work or was not installed properly, or there is a problem with the **disk** drive itself.
1-2-3 RELEASE 1A
LOCATING CIRCULAR
REFERENCES
Locating circular references can be...

29/3,K/17 (Item 1 from file: 553)
DIALOG(R)File 553:Wilson Bus. Abs. FullText

03795636 H.W. WILSON RECORD NUMBER: BWBA98045636 (USE FORMAT 7 FOR FULLTEXT)
Windows utilities: Rx for your PC.
AUGMENTED TITLE: 28 products evaluated
Andrews, Dean
McCracken, Harry; Spector, Lincoln
PC World v. 16 no6 (June 1998) p. 110-13+
LANGUAGE: English
WORD COUNT: 7178

(USE FORMAT 7 FOR FULLTEXT)

TEXT:
... 98 will include the troubleshooting System Configuration Utility; the Version Conflict Manager, which prevents saving **old** files over **new** ; and the System File Checker. Add/Remove Programs will **remain** , but **Disk** Cleanup will empty the Recycle Bin and Internet **cache** . Win 98 will also **update** Win 3.1's Dr. Watson diagnostics; the new Automatic Skip Driver Agent will watch...

29/3,K/18 (Item 2 from file: 553)
DIALOG(R)File 553:Wilson Bus. Abs. FullText

01530804 H.W. WILSON RECORD NUMBER: BWBA89030804
Put a positive lock on your data.
Miastkowski, Stan
Byte v. 14 (Feb. 1989) p. 100
LANGUAGE: English

...ABSTRACT: types SEAL followed by the name of the file. DataSentry

encrypts the file into a **new** file about half the size of the **original** . It then **writes** over the old file, safely eradicating any information that could **remain** on the **disk** . To restore the encrypted file, the user must reinsert the key. The basic DataSentry package...

29/3,K/19 (Item 1 from file: 635)
DIALOG(R)File 635:Business Dateline(R)

0398075 93-49545
IIT announces XtraDrive Lite
Chirico, Robin
Business Wire (San Francisco, CA, US) s1 p1
PUBL DATE: 930521
WORD COUNT: 539
DATELINE: Santa Clara, CA, US

TEXT:

...When XtraDrive is installed on a system, it is just as if a new hard **disk** has been installed in your computer. Your original hard **disk** is **unchanged** . All the files are still there, in their original locations, and they may be read and **written** just as before. In short, nothing has changed, except that there is a **new** , virtual **disk** on your system. The **original** disk is compressed but this compression is invisible to the programs, even those that circumvent...

29/3,K/20 (Item 1 from file: 636)
DIALOG(R)File 636:Gale Group Newsletter DB(TM)

01967048 Supplier Number: 43499702 (USE FORMAT 7 FOR FULLTEXT)
GOVERNMENT AGENCIES ENTER NEW AGE OF INFORMATION MANAGEMENT WITH OPTICAL IMAGING
Electronic Imaging Report, v2, n24, pN/A
Dec 2, 1992
Language: English Record Type: Fulltext
Document Type: Magazine/Journal; Trade
Word Count: 1338

... user, however, write-once systems appear to operate just as Winchester drives--information can be **updated** and edited, files can be deleted. Each new version is saved to a **new** location on the **disk** ; the **original** file **remains** intact and is accessible with special utilities.
This security feature has made **write** -once technology popular for a growing number of applications. For example, many election officials throughout...

29/3,K/21 (Item 1 from file: 810)
DIALOG(R)File 810:Business Wire

0818416 BW0041

QUARTERDECK CORP 2: Quarterdeck Introduces DiskClone Extra Strength; the

Fast and Easy Way to Upgrade and Copy Hard Disk Drives

March 09, 1998

Byline: Business & Technology Editors

...prior to copying, by
supporting FAT 32, and allowing partitions larger than 2.1 gigabytes.
DiskClone Extra Strength Features
-- Automatic **Adjustment** of Partitions
Copies images onto larger hard drives without requiring the drive
to be partitioned. **DiskClone** divides the **new** drive into the same
proportions as the **original** , copied drive.
-- Compression
Compresses data prior to imaging, providing faster **transfer** rates
and convenient storage.
-- FAT 32 Support
Allows users of Windows 95 or 98 to...

29/3,K/22 (Item 2 from file: 810)
DIALOG(R)File 810:Business Wire

0818412 BW0039

QUARTERDECK CORP: Quarterdeck Introduces DiskClone Corporate for Small and Medium-Sized Businesses; The Fast and Safe Way to Upgrade Operating Systems and Copy Hard Disk Drives Over a Network

March 09, 1998

Byline: Business Editors & Computer Writers

...1 gigabytes. Quarterdeck's
Partition-It Extra Strength, with Select-It, is also included in
DiskClone Corporate.
DiskClone Corporate Features
Scripting
Allows multiple installs over the network.
Automatic **Adjustment** of Partitions
Copies images onto larger hard drives without requiring
the drive to be partitioned. **DiskClone** divides the **new** drive into
the same proportions as the **original** , copied drive.
Compression
Compresses data prior to imaging, providing faster **transfer** rates and
convenient storage.
FAT 32 Support
Allows users of Windows 95 or 98 to...

29/3,K/23 (Item 1 from file: 813)
DIALOG(R)File 813:PR Newswire

0274085 NY027
PANASONIC DEBUTS MULTI-FUNCTION OPTICAL DRIVE WITH DIRECT OVERWRITE FEATURE

DATE: June 4, 1990 10:50 EDT WORD COUNT: 692

...overwrite information in one pass directly onto an optical disk," said Joe Videtti, national marketing **manager** , Panasonic Office Automation Group's **computer** systems division.

"Magneto-optical **disk** drives employ two passes over the optical media to first erase and then rewrite," Videtti continued. "The new phase-change technology that Panasonic uses permits **new** information to be **written** over **old** data within the same pass using a single laser beam.

This

Patents Full text files

```
? ds; show files

Set     Items   Description
S1     549282   (CACHE? OR   TEMPORARY()(STORAGE OR  MEMORY? )
             OR BUFFER? OR RAM?)
S2     963474   (WRIT??? OR WRIT???()DISK? OR TRANSFER??? OR RECORD??? OR -
             ENCOD??? OR UPDAT???)
S3      45900   (OLD OR ORIGINAL? OR PREVIOUS OR EARL??? OR OUTDAT???)(7N)-
             (NEW??? OR RECENT OR MODERN OR FRESH)
S4     150099   (MODIF???  OR MODIFICAT??? OR AMEND??? OR AMENDM??? OR CHA-
             NG??? OR ADJUST??? OR ADJUSTM???)(3N)(COP??? OR DATA OR FILE??
              OR INFORMATION?? OR INFO? OR VERSION? )
S5     669416   (NODE? OR TERMINAL? OR COMPUTER? OR CLIENT? OR SERVER? OR -
             WORKSTATION?? OR STATION??)
S6      25058   (PERSIST??? OR PERSEVER??? OR ENDUR??? OR LINGER??? OR REM-
             AIN??? OR STEAD??? OR PROLONG??? OR UNCHANG???)(3N)(STOR??? OR
              STORAGE? OR MEMOR??? OR HARD()DRIVE OR DISK? OR CD()ROM?? OR
             CDROM OR ROM?? OR FLASH? OR EPROM?? OR PROM??)
S7     176595   (MASTER? OR CONTROL??? OR MAIN OR PRIMARY? OR SUPERVIS??? -
              OR ADMINISTRA??? OR MANAG?)(5N) S5
S8         30   AU=(CHANDRASEKARAN, S?  OR CHANDRASEKARAN S?)
S9         30   AU=(BAMFORD, R? OR BAMFORD R? )
S10         9   AU=(BRIDGE, W? OR  BRIDGE W?)
S11        27   AU=(BROWER, D? OR  BROWER D?)
S12         6   AU=(MACNAUGHTON, N?  OR MACNAUGHTON N?)
S13       170   AU=(CHAN, W? OR  CHAN W?)
S14         5   AU=(SRIHARI, V? OR SRIHARI V?)
S15       243   S8 OR S9 OR S10 OR S11 OR S12 OR S13 OR S14
S16         4   S8 AND S9 AND S10 AND S11 AND S12 AND S13 AND S14
S17        20   S1(S)S2(S)S3(S)S4(S)S6(S)S7
S18      1914   S1(S)S2(S)S3
S19       239   S1(S)S2(S)S3(S)S4
S20        51   S1(10N)S2(10N)S3(10N)S4
S21        66   S1(S)S2(S)S3(S)S6
S22       167   S1(S)S2(S)S3(S)S7
S23       553   S1(S)S2(S)S3(S)S5
S24        67   S1(25N)S2(25N)S3(25N)S7
S25       168   S1(10N)S2(10N)S3(10N)S5
S26    136184   IC=G06F?
S27       297   S17 OR S20 OR S21 OR S24 OR S25
S28       175   S27 AND S26
S29        78   S28 NOT PY>1999
S30       110   S1(7N)S2(7N)S3(7N)S5
S31        46   S1(20N)S2(20N)S3(20N)S7
S32       236   S17 OR S20 OR S21 OR S30 OR S31
S33        37   S1(7N)S2(7N)S3(7N)S4
S34        14   S1(10N)S2(10N)S3(10N)S6
S35        37   S1(3N)S2(3N)S3(3N)S5
S36       137   S17 OR S33 OR S34 OR S35 OR S31
S37        92   S36 AND S26
S38        48   S37 NOT PY>1999
S39        39   S38 NOT PD=19981124:20001124
S40        39   S39 NOT PD=20001124:20021124
S41        39   S40 NOT PD=20021124:20050103
S42        39   IDPAT (sorted in duplicate/non-duplicate order)
S43        38   IDPAT (primary/non-duplicate records only)
File 348:EUROPEAN PATENTS 1978-2004/Dec W03
         (c) 2004 European Patent Office
File 349:PCT FULLTEXT 1979-2002/UB=20041230,UT=20041223
         (c) 2004 WIPO/Univentio
```

16/3,K/1 (Item 1 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

01498400
MANAGING CHECKPOINT QUEUES IN A MULTIPLE NODE SYSTEM
VERWALTUNG VON PRUFPUNKTSCHLANGEN IN EINEM MEHRKNOTENSYSTEM
GESTION DE FILES D'ATTENTE DE POINT DE CONTROLE DANS UN SYSTEME A NOEUDS MULTIPLES
PATENT ASSIGNEE:
Oracle International Corporation, (4124510), 500 Oracle Parkway, Redwood Shores, CA 94065, (US), (Applicant designated States: all)
INVENTOR:
CHANDRASEKARAN, Sashikanth , 2515 Carlmont Drive 9, Belmont, CA 94002, (US)
BAMFORD, Roger , 555 Manzanita Way, Woodside, CA 94062, (US)
BRIDGE, William , 2969 Seaview Parkway, Alameda, CA 94502, (US)
BROWER, David , 290 Livorna Heights Road, Alamo, CA 94507, (US)
MACNAUGHTON, Neil , 347 Penny Avenue, Los Gatos, CA 95030, (US)
CHAN, Wilson , 129 Woodbridge Circle, San Mateo, CA 94403, (US)
SRIHARI, Vinay , 7 BayCrest Way, San Francisco, CA 94080, (US
LEGAL REPRESENTATIVE:
Viering, Jentschura & Partner (100645), Postfach 22 14 43, 80504 Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 1412858 A2 040428 (Basic)
WO 2002073416 020919
APPLICATION (CC, No, Date): EP 2002717580 020307; WO 2002US6981 020307
PRIORITY (CC, No, Date): US 274270 P 010307; US 92247 020304
DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI; LU; MC; NL; PT; SE; TR
EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI
INTERNATIONAL PATENT CLASS: G06F-012/00
NOTE:
No A-document published by EPO
LANGUAGE (Publication,Procedural,Application): English; English; English

INVENTOR:
CHANDRASEKARAN, Sashikanth ...

...US)
BAMFORD, Roger ...

...US)
BRIDGE, William ...

...US)
BROWER, David ...
...US)
MACNAUGHTON, Neil ...

...US)
CHAN, Wilson ...

...US)
SRIHARI, Vinay ...

16/3,K/2 (Item 2 from file: 348)

DIALOG(R)File 348:EUROPEAN PATENTS

01491351
DISK WRITES IN A DISTRIBUTED SHARED DISK SYSTEM
DISKRETE SCHREIBVORGANGE IN EINEM VERTEILTEN SYSTEM MIT GEMEINSAM BENUTZTER PLATTE
ECRITURE SUR DISQUE DANS UN SYSTEME DE DISQUES COMMUNS REPARTIS
PATENT ASSIGNEE:
Oracle International Corporation, (4124510), 500 Oracle Parkway, Redwood Shores, CA 94065, (US), (Applicant designated States: all)
INVENTOR:
CHANDRASEKARAN, Sashikanth , 2515 Carlmont Drive 9, Belmont, CA 94002, (US)
BAMFORD, Roger , 555 Manzanita Way, Woodside, CA 94062, (US)
BRIDGE, William , 2969 Seaview Parkway, Alameda, CA 94502, (US)
BROWER, David , 290 Livorna Heights Road, Alamo, CA 94507, (US)
MACNAUGHTON, Neil , 347 Penny Avenue, Los Gatos, CA 95030, (US)
CHAN, Wilson , 129 Wodbridge Circle, San Mateo, CA 94403, (US)
SRIHARI, Vinay , 7 BayCrest Way, San Francisco, CA 94080, (US
LEGAL REPRESENTATIVE:
Viering, Jentschura & Partner (100645), Postfach 22 14 43, 80504 Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 1366420 A2 031203 (Basic)
WO 2002071229 020912
APPLICATION (CC, No, Date): EP 2002748408 020306; WO 2002US7475 020306
PRIORITY (CC, No, Date): US 274270 P 010307
DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI; LU; MC; NL; PT; SE; TR
EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI
INTERNATIONAL PATENT CLASS: G06F-012/08
NOTE:
No A-document published by EPO
LANGUAGE (Publication,Procedural,Application): English; English; English

INVENTOR:
CHANDRASEKARAN, Sashikanth ...

...US)
BAMFORD, Roger ...

...US)
BRIDGE, William ...

...US)
BROWER, David ...
...US)
MACNAUGHTON, Neil ...

...US)
CHAN, Wilson ...

...US)
SRIHARI, Vinay ...

16/3,K/3 (Item 1 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00939283 **Image available**
MANAGING CHECKPOINT QUEUES IN A MULTIPLE NODE SYSTEM
GESTION DE FILES D'ATTENTE DE POINT DE CONTROLE DANS UN SYSTEME A NOEUDS MULTIPLES
Patent Applicant/Assignee:
ORACLE INTERNATIONAL CORPORATION, 500 Oracle Parkway, Redwood Shores, CA 94065, US, US (Residence), US (Nationality)
Inventor(s):
CHANDRASEKARAN Sashikanth , 2515 Carlmont Drive #9, Belmont, CA 94002, US,
BAMFORD Roger , 555 Manzanita Way, Woodside, CA 94062, US,
BRIDGE William , 2969 Seaview Parkway, Alameda, CA 94502, US,
BROWER David , 290 Livorna Heights Road, Alamo, CA 94507, US,
MACNAUGHTON Neil , 347 Penny Avenue, Los Gatos, CA 95030, US,
CHAN Wilson , 129 Woodbridge Circle, San Mateo, CA 94403, US,
SRIHARI Vinay , 7 BayCrest Way, San Francisco, CA 94080, US
Legal Representative:
HICKMAN Brian (agent), HICKMAN PALERMO TRUONG & BECKER, LLP, 1600 Willow Street, San Jose, CA 95125, US,
Patent and Priority Information (Country, Number, Date):
Patent: WO 200273416 A2-A3 20020919 (WO 0273416)
Application: WO 2002US6981 20020307 (PCT/WO US02006981)
Priority Application: US 2001274270 20010307; US 200292247 20020304
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ OM PH PL PT RO RU SD SE SG SI SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA ZM ZW
(EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE TR
(OA) BF BJ CF CG CI CM GA GN GQ GW ML MR NE SN TD TG
(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZM ZW
(EA) AM AZ BY KG KZ MD RU TJ TM
Publication Language: English
Filing Language: English
Fulltext Word Count: 16774

Inventor(s):
CHANDRASEKARAN Sashikanth ...

... **BAMFORD Roger** ...
... **BRIDGE William** ...

... **BROWER David** ...

... **MACNAUGHTON Neil** ...

... **CHAN Wilson** ...

... **SRIHARI Vinay**

16/3,K/4 (Item 2 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00937121 **Image available**
DISK WRITES IN A DISTRIBUTED SHARED DISK SYSTEM
ECRITURE SUR DISQUE DANS UN SYSTEME DE DISQUES COMMUNS REPARTIS
Patent Applicant/Assignee:
ORACLE INTERNATIONAL CORPORATION, 500 Oracle Parkway, Redwood Shores, CA 94065, US, US (Residence), US (Nationality)
Inventor(s):
CHANDRASEKARAN Sashikanth , 2515 Carlmont Drive #9, Belmont, CA 94002, US,
BAMFORD Roger , 555 Manzanita Way, Woodside, CA 94062, US,
BRIDGE William , 2969 Seaview Parkway, Alameda, CA 94502, US,
BROWER David , 290 Livorna Heights Road, Alamo, CA 94507, US,
MACNAUGHTON Neil , 347 Penny Avenue, Los Gatos, CA 95030, US,
CHAN Wilson , 129 Wodbridge Circle, San Mateo, CA 94403, US,
SRIHARI Vinay , 7 BayCrest Way, San Francisco, CA 94080, US
Legal Representative:
HICKMAN Brian (et al) (agent), HICKMAN PALERMO TRUONG & BECKER, LLP, 1600 Willow Street, San Jose, CA 95125, US,
Patent and Priority Information (Country, Number, Date):
Patent: WO 200271229 A2-A3 20020912 (WO 0271229)
Application: WO 2002US7475 20020306 (PCT/WO US0207475)
Priority Application: US 2001274270 20010307; US 20021618 20020304
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ
EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR
LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ OM PH PL PT RO RU SD SE SG SI
SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA ZM ZW
(EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE TR
(OA) BF BJ CF CG CI CM GA GN GQ GW ML MR NE SN TD TG
(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZM ZW
(EA) AM AZ BY KG KZ MD RU TJ TM
Publication Language: English
Filing Language: English
Fulltext Word Count: 19733

Inventor(s):
CHANDRASEKARAN Sashikanth ...

... **BAMFORD Roger** ...

... **BRIDGE William** ...
... **BROWER David** ...

... **MACNAUGHTON Neil** ...

... **CHAN Wilson** ...

... **SRIHARI Vinay**
?

43/3,K/1 (Item 1 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00833812
Method and apparatus for reliable disk fencing in a multicomputer system
Verfahren und Vorrichtung zum zuverlassigen Ausgrenzen von fehlerhaften Plattenspeicherbereichen in einem Mehrrechnersystem
Methode et dispositif pour une delimitation fiable des zones erronees de disques dans un systeme multiordinateur
PATENT ASSIGNEE:
SUN MICROSYSTEMS, INC., (1392730), 2550 Garcia Avenue, Mountain View, CA 94043, (US), (applicant designated states: DE;FR;GB;NL;SE)
INVENTOR:
Matena, Vladimir, 1322 Kentfield Avenue, Redwood City, California 94061, (US)
LEGAL REPRESENTATIVE:
Hogg, Jeffery Keith et al (31905), Withers & Rogers 4 Dyer's Buildings Holborn, London EC1N 2JT, (GB)
PATENT (CC, No, Kind, Date): EP 772126 A2 970507 (Basic)
APPLICATION (CC, No, Date): EP 96307676 961023;
PRIORITY (CC, No, Date): US 552316 951102
DESIGNATED STATES: DE; FR; GB; NL; SE
INTERNATIONAL PATENT CLASS: **G06F-011/00** ; **G06F-009/46** ; **G06F-015/16**
ABSTRACT WORD COUNT: 193

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB97	649
SPEC A	(English)	EPAB97	4419
Total word count - document A			5068
Total word count - document B			0
Total word count - documents A + B			5068

INTERNATIONAL PATENT CLASS: **G06F-011/00** ...

... **G06F-009/46** ...

... **G06F-015/16**

...SPECIFICATION g. a failed node), it is not allowed to change its NK. If such a **node** attempts to read or **write** to a **disk** , the controller finds a mismatch between the **new** CK value and the **old** NK value.
When a **node** is started, its NK is initialized to a 0 value.

Procedures for Calculating Values of...

43/3,K/2 (Item 2 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00695231
An improved data storage device and method of operation.
Ein verbesserter Datenspeichergerat und Betriebsverfahren hierzu.
Un dispositif de stockage de donnees ameliore et methode d'operation.
PATENT ASSIGNEE:

INTERNATIONAL BUSINESS MACHINES CORPORATION, (200125), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: AT;BE;CH;DE;ES;FR;GB;IT;LI;NL;SE)
INVENTOR:
Bandy, Peter Burghardt, P.O. Box 380A, Pine Island, Minnesota 55913, (US)
LEGAL REPRESENTATIVE:
Therias, Philippe (77261), Compagnie IBM FRANCE, Departement de Propriete Intellectuelle, F-06610 La Gaude, (FR)
PATENT (CC, No, Kind, Date): EP 662660 A1 950712 (Basic)
APPLICATION (CC, No, Date): EP 94480163 941206;
PRIORITY (CC, No, Date): US 178955 940107
DESIGNATED STATES: AT; BE; CH; DE; ES; FR; GB; IT; LI; NL; SE
INTERNATIONAL PATENT CLASS: **G06F-011/10** ; G11B-020/18
ABSTRACT WORD COUNT: 145

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB95	619
SPEC A	(English)	EPAB95	3357
Total word count - document A			3976
Total word count - document B			0
Total word count - documents A + B			3976

INTERNATIONAL PATENT CLASS: **G06F-011/10** ...

...SPECIFICATION RAID technology. In it, they describe RAID Level 5 as a partial solution to the **write** parity bottleneck that may occur in RAID Level 4. RAID Level 4 is analogous to...

...to the host computer by the SCSI bus, and the array is controlled by Array **Management** Software operating in the host **computer** . The known alternative of providing an I/O controller is more expensive and, as described...

...a bottleneck. The bottleneck is accentuated by the fact that the host must either send **write** commands to all of the **disks** , as in RAID Levels 2 and 3, or must read the old data and the **old** parity in order to generate **new** parity for new data and then **write** the new data and new parity as in RAID Levels 4 and 5.
SUMMARY OF...

...means in each storage device in the array. According to the invention, data to be **written** is sent to the target drive from the host computer under a new command that...

43/3,K/3 (Item 3 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00682352
Cache control system for managing validity status of data stored in a cache memory.
Cachespeichersteuerungssystem zur Verwaltung des Gultigkeitzustands von im Cachespeicher gespeicherten Daten.
Systeme de controle d'une antememoire pour la gestion d'etat de validite de donnees stockees dans l'antememoire.

PATENT ASSIGNEE:
INTERNATIONAL BUSINESS MACHINES CORPORATION, (200125), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: DE;FR;GB)
INVENTOR:
Breternitz, Mauricio, Dr. Jnr., 5714 Penny Creek Drive, Austin, Texas 78759, (US)
LEGAL REPRESENTATIVE:
Lloyd, Richard Graham (75501), IBM (UK) Ltd, UK Intellectual Property Department, Hursley Park, Winchester, Hampshire SO21 2JN, (GB)
PATENT (CC, No, Kind, Date): EP 652520 A1 950510 (Basic)
APPLICATION (CC, No, Date): EP 94308133 941104;
PRIORITY (CC, No, Date): US 150662 931109
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-012/08**
ABSTRACT WORD COUNT: 196

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB95	816
SPEC A	(English)	EPAB95	2826
Total word count - document A			3642
Total word count - document B			0
Total word count - documents A + B			3642

INTERNATIONAL PATENT CLASS: **G06F-012/08**

...SPECIFICATION at the processor cycle time in order to optimize CPU performance by improving effective data **transfer** rates. Cache memory comprises a small, relatively fast access memory which is interposed between a larger, relatively slow main system memory. In many applications, the existence of **cache** memory is essentially hidden or "transparent" to both the CPU and the data processing system user.

Typically, **cache** memory is managed by control logic which is designed to load the **cache** memory with information which is most frequently accessed by the CPU as directed by a **computer** application program. The **cache** memory **control** logic must also make space available for **new** information or clear out **old** information which is no longer needed by the application program. Therefore, **cache** management procedures necessarily include selectively invalidating rows of data in the **cache** memory.

This process of making **new** memory space available and clearing out **old** information by selectively invalidating rows of data in **cache** memory may be referred to as maintaining **cache** coherence. The process of maintaining cache coherence may be implemented in hardware, however, this solution...

43/3,K/4 (Item 4 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00674691
Data storage system architecture
Datenspeicherungssystemarchitektur
Architecture de systeme de stockage de donnees
PATENT ASSIGNEE:
NCR International, Inc., (1449484), 1700 South Patterson Boulevard,

Dayton, Ohio 45479, (US), (applicant designated states: DE;FR;GB)
HYUNDAI ELECTRONICS AMERICA, (1312771), 510 Cottonwood Drive, Milpitas, California 95035, (US), (applicant designated states: DE;FR;GB)
SYMBIOS LOGIC INC., (1907320), 2001 Danfield Court, Fort Collins, Colorado 80525, (US), (applicant designated states: DE;FR;GB)
INVENTOR:
DuLac, Keith Bernard, 8652 Hila, Derby, KS 67037, (US)
LEGAL REPRESENTATIVE:
Gill, David Alan et al (69772), W.P. Thompson & Co., Celcon House, 289-293 High Holborn, London WC1V 7HU, (GB)
PATENT (CC, No, Kind, Date): EP 646858 A1 950405 (Basic)
EP 646858 B1 980415
APPLICATION (CC, No, Date): EP 94306322 940826;
PRIORITY (CC, No, Date): US 124653 930907
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-003/06** ; **G06F-013/40** ; **G06F-011/20**
ABSTRACT WORD COUNT: 150

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9816	585
CLAIMS B	(German)	9816	617
CLAIMS B	(French)	9816	680
SPEC B	(English)	9816	2277
Total word count - document A			0
Total word count - document B			4159
Total word count - documents A + B			4159

INTERNATIONAL PATENT CLASS: **G06F-003/06** ...

... **G06F-013/40** ...

... **G06F-011/20**

...SPECIFICATION is in buffer. Node (0,1) reads old parity data over column bus C1)) into **node** (0,1) **buffer** B2. **Node** (0,1) now has **new** data, **old** data and **old** parity in its **buffer** .

To complete the RAID 5 **write** operation, **node** processor (0,1) orders an exclusive-OR of the data stored within buffers B1, B2...

43/3,K/5 (Item 5 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00668699
Parallel database system.
Paralleles Datenbanksystem.
Systeme de base de donnees parallele.
PATENT ASSIGNEE:
INTERNATIONAL BUSINESS MACHINES CORPORATION, (200125), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: DE;FR;GB)
INVENTOR:
Li, Shih-Gong, 9402 Mystic Oaks Trail, Austin, Texas 78750, (US)
LEGAL REPRESENTATIVE:
Davies, Simon Robert (75451), I B M UK Intellectual Property Department Hursley Park, Winchester, Hampshire SO21 2JN, (GB)
PATENT (CC, No, Kind, Date): EP 642092 A2 950308 (Basic)

EP 642092 A3 950809
APPLICATION (CC, No, Date): EP 94306365 940830;
PRIORITY (CC, No, Date): US 116089 930902
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-017/30**
ABSTRACT WORD COUNT: 187

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB95	401
SPEC A	(English)	EPAB95	6031
Total word count - document A			6432
Total word count - document B			0
Total word count - documents A + B			6432

INTERNATIONAL PATENT CLASS: **G06F-017/30**

...SPECIFICATION PDB system, "buckets" of data must be moved from the existing nodes to the new **nodes** . A logical link is established with a predefined number of communication **buffers** for sending data **records** from the **old** residing **node** to the **new** **node** . As most relational database systems do not support a physical bucket in their storage organization...

43/3,K/6 (Item 6 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00600342
System and methods for file management.
System und Verfahren zur Dateiverwaltung.
Systeme et procedes de gestion de fichiers.
PATENT ASSIGNEE:
BORLAND INTERNATIONAL, Inc., (1423872), 1800 Green Hills Road, Scotts Valley, California 95041, (US), (applicant designated states: DE;FR;GB;IE;IT)
INVENTOR:
Shaughnessy, Steven T., P.O. Box 613, Mt. Hermon, California 95041, (US)
LEGAL REPRESENTATIVE:
Godsill, John Kenneth et al (31031), Haseltine Lake & Co. Hazlitt House 28 Southampton Buildings Chancery Lane, London WC2A 1AT, (GB)
PATENT (CC, No, Kind, Date): EP 588502 A2 940323 (Basic)
EP 588502 A3 960131
APPLICATION (CC, No, Date): EP 93306518 930818;
PRIORITY (CC, No, Date): US 933480 920820
DESIGNATED STATES: DE; FR; GB; IE; IT
INTERNATIONAL PATENT CLASS: **G06F-015/40**
ABSTRACT WORD COUNT: 197
LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF2	1125
SPEC A	(English)	EPABF2	12193
Total word count - document A			13318
Total word count - document B			0
Total word count - documents A + B			13318

INTERNATIONAL PATENT CLASS: **G06F-015/40**

...SPECIFICATION to the various family members based on the concurrency information read. Moreover, the logical lock **file** itself may be **modified** , including adding **new** registration entries and deleting **old** ones; in this instance, the **new** file is **written** back to the storage **disk** , with the size **written** being remembered as the current size. The method loops to selected ones of the above...

43/3,K/7 (Item 7 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00599104
Computer system maintaining data consistency between the cache and the main memory.
Computersystem mit Aufrechterhaltung der Datenubereinstimmung zwischen dem Cache-Speicher und dem Hauptspeicher.
Systeme de traitement d'information permettant de maintenir la coherence des donnees entre l'entememoire et la memoire principale.
PATENT ASSIGNEE:
INTERNATIONAL BUSINESS MACHINES CORPORATION, (200125), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: DE;FR;GB)
INVENTOR:
Oba, Nobuyuki, 2-8-6-201 Dainohara Aoba-ku, Sendai-shi, Miyagi-ke, (JP)
Shimizu, Shigenori, 4-29-1-406 Kamiasao, Asao-ku, Kwasaki-shi, Kanagawa-ken, (JP)
LEGAL REPRESENTATIVE:
Burt, Roger James, Dr. (52152), IBM United Kingdom Limited Intellectual Property Department Hursley Park, Winchester Hampshire SO21 2JN, (GB)
PATENT (CC, No, Kind, Date): EP 579418 A2 940119 (Basic)
EP 579418 A3 950118
APPLICATION (CC, No, Date): EP 93305156 930701;
PRIORITY (CC, No, Date): JP 92175398 920702
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-012/08**
ABSTRACT WORD COUNT: 153

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF2	217
SPEC A	(English)	EPABF2	3543
Total word count - document A			3760
Total word count - document B			0
Total word count - documents A + B			3760

INTERNATIONAL PATENT CLASS: **G06F-012/08**

...SPECIFICATION and the system bus (off-chip cache).
In cases when a cache memory is provided, **cache** consistency will be the major problem. In other words, to put it simply, the problem is whether or not the data contained in the **cache** memory is always the latest. If **old** data **remains** in the **cache** **memory** when **new** data is **written** in the main memory from outside, the microprocessor reads that old data and proper operations...

43/3,K/8 (Item 8 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00590481
Redundant array of disks with improved storage and recovery speed
Redundante Speicherplattenmatrix mit verbesserter Schreib- und Lesegeschwindigkeit
Reseau redondant de disques avec une vitesse d'ecriture et de lecture amelioree
PATENT ASSIGNEE:
MITSUBISHI DENKI KABUSHIKI KAISHA, (208580), 2-3, Marunouchi 2-chome Chiyoda-ku, Tokyo 100, (JP), (applicant designated states: DE;FR;GB)
INVENTOR:
Matsumoto, Toshio, c/o Mitsubishi Denki K.K., Computer Seisakusho, 325 Kamimachiya, Kamakura-shi, Kanagawa 247, (JP)
Baba, Hiroshi, c/o Mitsubishi Denki K.K., Computer Seisakusho, 325 Kamimachiya, Kamakura-shi, Kanagawa 247, (JP)
Itoh, Kazuhiko, c/o Mitsubishi Denki K.K., Computer Seisakusho, 325 Kamimachiya, Kamakura-shi, Kanagawa 247, (JP)
Ogura, Shiro, c/o Mitsubishi Denki K.K., Computer Seisakusho, 325 Kamimachiya, Kamakura-shi, Kanagawa 247, (JP)
LEGAL REPRESENTATIVE:
Pfenning, Meinig & Partner (100961), Mozartstrasse 17, 80336 Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 584804 A2 940302 (Basic)
EP 584804 A3 940810
EP 584804 B1 981104
APPLICATION (CC, No, Date): EP 93113563 930825;
PRIORITY (CC, No, Date): JP 92226976 920826
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: G11B-020/12; G11B-020/18; **G06F-003/06**
ABSTRACT WORD COUNT: 122

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9845	638
CLAIMS B	(German)	9845	639
CLAIMS B	(French)	9845	776
SPEC B	(English)	9845	9354
Total word count - document A			0
Total word count - document B			11407
Total word count - documents A + B			11407

...INTERNATIONAL PATENT CLASS: **G06F-003/06**
...SPECIFICATION interval to see if further commands to store data will be received from the host **computer** before proceeding to the tasks of reading **old** data from the **disks** , computing **new** check information, and **writing** the new check information. This interval can be adjusted to obtain a desired trade-off...

43/3,K/9 (Item 9 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00491269
Storage device array architecture with copyback cache
Speichergeratanordnungsarchitektur mit Copyback-Cachespeicher
Architecture d'un reseau de dispositif de memoire a antememoire a post-memorisation
PATENT ASSIGNEE:
EMC CORPORATION, (1739001), 171 South Street, Hopkinton, MA 01748-9103, (US), (applicant designated states: DE;FR;GB;IT)
INVENTOR:
Brant, William Alexander, 4784 Dorchester Circle, Boulder Colorado 80301, (US)
Walker, Mark, 20,000 Gist Road, Los Gatos California 95030, (US)
Stallmo, David Charles, 59 Beaver Way, Boulder Colorado 80304, (US)
Lui, Albert, 3164 Heritage Valley Drive, San Jose California 95148, (US)
LEGAL REPRESENTATIVE:
Driver, Virginia Rozanne et al (58902), Page White & Farrer 54 Doughty Street, London WC1N 2LS, (GB)
PATENT (CC, No, Kind, Date): EP 493984 A2 920708 (Basic)
EP 493984 A3 930512
EP 493984 B1 970604
APPLICATION (CC, No, Date): EP 91312104 911231;
PRIORITY (CC, No, Date): US 638167 910104
DESIGNATED STATES: DE; FR; GB; IT
INTERNATIONAL PATENT CLASS: **G06F-011/10** ; G11B-020/18; **G06F-012/08**
ABSTRACT WORD COUNT: 263

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPAB97	1224
CLAIMS B	(German)	EPAB97	1061
CLAIMS B	(French)	EPAB97	1641
SPEC B	(English)	EPAB97	5609
Total word count - document A			0
Total word count - document B			9535
Total word count - documents A + B			9535

INTERNATIONAL PATENT CLASS: **G06F-011/10** ...

... **G06F-012/08**

...SPECIFICATION location in the array is still a "pending block" (a data block that has been **Written** to the copyback **cache** storage unit CC but not **transferred** to the array S1-S5), the directory location pointer for the **data** block is **changed** to point to the " **new** " version rather than to the " **old** " version. The old version is thereafter ignored, and may be written over in subsequent operations...at the same location in the array is still a "pending block" in the controller **buffer** , the directory location pointers for the **data** block are **changed** to point to the " **new** " version rather than to the " **old** " version both in the copyback **cache** storage unit CC and in the **buffer** . The old version is thereafter ignored, and may be **written** over in subsequent operations.

After a **Write** request is processed in this fashion, the controller 3 immediately sends an acknowledgement to the...

43/3,K/10 (Item 10 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00489592
Rotating memory system
Rotierende Speicheranordnung
Systeme de memoire rotative
PATENT ASSIGNEE:
INTERNATIONAL COMPUTERS LIMITED, (233330), ICL House, Putney, London, SW15 1SW, (GB), (applicant designated states: BE;DE;FR;GB;IT)
INVENTOR:
Holt, Nicholas Peter, 104 Padfield Main Road, Padfield, Hadfield, via Hyde, Cheshire SK14 7ET, (GB)
LEGAL REPRESENTATIVE:
Guyatt, Derek Charles et al (31321), Intellectual Property Department International Computers Limited Cavendish Road, Stevenage, Herts, SG1 2DY, (GB)
PATENT (CC, No, Kind, Date): EP 490485 A2 920617 (Basic)
EP 490485 A3 941214
EP 490485 B1 970514
APPLICATION (CC, No, Date): EP 91310390 911111;
PRIORITY (CC, No, Date): GB 9026917 901211
DESIGNATED STATES: BE; DE; FR; GB; IT
INTERNATIONAL PATENT CLASS: **G06F-003/06** ; G11B-019/02; G11B-005/012
ABSTRACT WORD COUNT: 86

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF1	210
CLAIMS B	(English)	EPAB97	297
CLAIMS B	(German)	EPAB97	288
CLAIMS B	(French)	EPAB97	344
SPEC A	(English)	EPABF1	2047
SPEC B	(English)	EPAB97	2188
Total word count - document A			2257
Total word count - document B			3117
Total word count - documents A + B			5374

INTERNATIONAL PATENT CLASS: **G06F-003/06** ...

...SPECIFICATION commit the transaction, by deallocating the before-look blocks, or abort it, by deallocating the **newly** **written** blocks and restoring the **previous** mapping.

Disk memory systems interfaced using SCSI (Small **computer** system interface) have an undefined internal layout in terms of tracks, cylinders etc and all...

...SPECIFICATION commit the transaction, by deallocating the before-look blocks, or abort it, by deallocating the **newly** **written** blocks and restoring the **previous** mapping.

Disk memory systems interfaced using SCSI (Small **computer** system interface) have an undefined internal layout in terms of tracks, cylinders etc and all...

43/3,K/11 (Item 11 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00482697
File-based transaction management system for a computing system.
Dateibasiertes Transaktionsverwaltungssystem fur ein Rechnersystem.
Systeme de gestion des transactions basees sur fichiers dans un systeme de calcul.
PATENT ASSIGNEE:
International Business Machines Corporation, (200120), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: DE;FR;GB)
INVENTOR:
Frey, Alexander H., Jr., 1570 Rose Villa Street, Pasadena, CA 91106, (US)
Mosteller, Richard C., 420 Lima Street, Sierra Madre, CA 91024, (US)
Gould, Joel M., 24 Prescott Road, Norwood, MA 02062, (US)
Mendelsohn, Noah R., 136 Thorndike Street, Arlington, MA 02174, (US)
Perchik, James, 295 Harvard Street, No.607, Cambridge, MA 02139, (US)
LEGAL REPRESENTATIVE:
Bailey, Geoffrey Alan et al (27921), IBM United Kingdom Limited Intellectual Property Department Hursley Park, Winchester Hampshire SO21 2JN, (GB)
PATENT (CC, No, Kind, Date): EP 454340 A2 911030 (Basic)
EP 454340 A3 930908
APPLICATION (CC, No, Date): EP 91303357 910416;
PRIORITY (CC, No, Date): US 509853 900416
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-011/14** ; **G06F-015/403** ; **G06F-015/40**
ABSTRACT WORD COUNT: 160

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF1	793
SPEC A	(English)	EPABF1	5650
Total word count - document A			6443
Total word count - document B			0
Total word count - documents A + B			6443

INTERNATIONAL PATENT CLASS: **G06F-011/14** ...

... **G06F-015/403** ...

... **G06F-015/40**

...SPECIFICATION in the transaction data field so that if a malfunction occurs prior to "new" or " **modified** " **data** being **written** to **disk** , information is available as to the **original** or " **old** " state of the **data** . The " **new** " or " **modified** " **data** also may be included in the transaction data field so that the transaction may be "committed" prior to the base **data** being **modified** on **disk** . Finally, the Tail is a small non-zero value which is used to verify that the entire log entry was **written** .

If a transaction involves more than one node, during processing time, the invoking node assembles...

43/3,K/12 (Item 12 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00451675

NETWORK SYSTEM CONSISTING OF TERMINALS HAVING DOCUMENT PREPARING FUNCTION.
NETZSYSTEM BESTEHEND AUS ENDSTATIONEN MIT DOKUMENTVORBEREITUNGSWIRKUNG.
SYSTEME EN RESEAU COMPOSE DE TERMINAUX POSSEDANT UNE FONCTION DE PREPARATION DE DOCUMENTS.

PATENT ASSIGNEE:
FUJITSU LIMITED, (211460), 1015, Kamikodanaka Nakahara-ku, Kawasaki-shi Kanagawa 211, (JP), (applicant designated states: DE;FR;GB)

INVENTOR:
RYU, Tadamitsu, 1-604, Konandai Kotohausu 1151-121, Kamigocho, Sakae-ku Yokohama-shi Kanagawa 247, (JP)
GAMOH, Mineo, 27-5, Uenashi, Yuki-shi Ibaragi 307, (JP)
TANIDA, Toshitsugu, 577, Oaza Tagawa, Oyama-shi Tochigi 307-02, (JP)
WATANABE, Toshiki, W-10-222 Fujitsu Nakaharahausu 426, Siboguchi, Takatsu-ku Kawasaki-shi Kanagawa 213, (JP)
UEDA, Kenichi, 13-1, Asahicho 1-chome, Kawaguchi-shi Saitama 332, (JP)
ISONO, Etsuko, 202, Sofare A 462-2, Ishibashi Ishibashicho, Simotsuga-gun Tochigi 329-05, (JP)
MOGI, Yoshio, 102-4, Hiraicho, Tochigi-shi Tochigi 328, (JP)
FUKATSU, Takanori, 101 Sankei Manshon 665, Miyazaki Miyamae-ku, Kawasaki-shi Kanagawa 213, (JP)
KITAO, Taiji, 101 c/o Wada 351-5, Chitose Takatsu-ku, Kawasaki-shi Kanagawa 213, (JP)

LEGAL REPRESENTATIVE:
Lehn, Werner, Dipl.-Ing. et al (7471), Hoffmann, Eitle & Partner Patentanwalte Arabellastrasse 4, W-8000 Munchen 81, (DE)

PATENT (CC, No, Kind, Date): EP 443041 A1 910828 (Basic)
EP 451282 A1 911016
EP 451282 A1 930203
WO 9104545 910404

APPLICATION (CC, No, Date): EP 90913533 900912; WO 90JP1169 900912
PRIORITY (CC, No, Date): JP 89236052 890912; JP 89236053 890912; JP 89270048 891017; JP 89297961 891116; JP 89339370 891227; JP 9099830 900416
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-015/72**
ABSTRACT WORD COUNT: 116

LANGUAGE (Publication,Procedural,Application): English; English; Japanese
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF1	1262
SPEC A	(English)	EPABF1	9120
Total word count - document A			10382
Total word count - document B			0
Total word count - documents A + B			10382

INTERNATIONAL PATENT CLASS: **G06F-015/72**

...CLAIMS according to claims 1, 2 and 3, wherein the program file (6) comprises a schedule **control** program and a **terminal** unit comprises a **buffer** for storing **old** schedule data already generated and the **newly** added or **updated** schedule data, generates a **new** schedule data by OR of the **old** schedule data and **newly** **updated** or added schedules and realizes graphic display

43/3,K/13 (Item 13 from file: 348)

00450840
BILLING SYSTEM
KOSTENRECHNUNGSSYSTEM
SYSTEME DE FACTURATION
PATENT ASSIGNEE:
  Centillion Data Systems, Inc., (2031950), 333 North Alabama Street, Indianapolis, Indiana 46204, (US), (applicant designated states: AT;BE;CH;DE;DK;ES;FR;GB;IT;LI;LU;NL;SE)
INVENTOR:
  HARDY, Robert, M., 3520 Carmel Drive, Carmel, IN 46032, (US)
  CAUFFMAN, John, M., 9093 Sweet Bay Court, Indianapolis, IN 46260, (US)
  CAUFFMAN, Lynn, S., 9093 Sweet Bay Court, Indianapolis, IN 46260, (US)
  Lovell, Robert C., 93 Christy Drive, Greenwood, New Jersey 46153, (US)
  FRAZIER, Murray, B., 4182-A Lake Park Boulevard, Indianapolis, IN 46227, (US)
  Johnson, Michael L., 5745 Cambrook Road, Dublin, Ohio 43017, (US)
  DOHREWEND, Jamers, W., Jr., 2579 Parr Drive, Indianapolis, IN 46220, (US)
LEGAL REPRESENTATIVE:
  Rau, Manfred, Dr. Dipl.-Ing. et al (38392), Rau, Schneck & Hubner Patentanwalte Konigstrasse 2, 90402 Nurnberg, (DE)
PATENT (CC, No, Kind, Date): EP 541535 A1 930519 (Basic)
EP 541535 A1 930901
EP 541535 B1 970709
WO 9103023 910307
APPLICATION (CC, No, Date): EP 90912250 900810; WO 90US4563 900810
PRIORITY (CC, No, Date): US 393699 890814
DESIGNATED STATES: AT; BE; CH; DE; DK; ES; FR; GB; IT; LI; LU; NL; SE
INTERNATIONAL PATENT CLASS: **G06F-017/60**
NOTE:
  No A-document published by EPO
LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPAB97	516
CLAIMS B	(German)	EPAB97	411
CLAIMS B	(French)	EPAB97	626
SPEC B	(English)	EPAB97	17633
Total word count - document A			0
Total word count - document B			19186
Total word count - documents A + B			19186

INTERNATIONAL PATENT CLASS: **G06F-017/60**
...SPECIFICATION invention (the "user application") to display and analyze their bill. When the user receives the **diskettes** , the information thereon must be decompressed and loaded into a PC database using facilities provided...

...available database software, such as "RBASE", a popular database package available for IBM-PC-compatible **computers** , to **manage** the billing **records** received on **diskette** . Except for a small amount of historical information used for certain graphs and summary reports, the database can contain only one "bill" at any time. When a **new** bill is received, the **previous** bill may be archived to a non-database file (flat file) on the user's **disk** for convenient retrieval. The **new** bill then replaces the **old** bill in the user application database.

When **writing** information into the database, the user application employs commercially available software routines, such as RBASE...

43/3,K/14 (Item 14 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00448075
APPARATUS AND METHOD FOR MAINTAINING CACHE/MAIN MEMORY CONSISTENCY
VORRICHTUNG UND VERFAHREN FUR INSTANDHALTUNG VON CACHE/ZENTRALSPEICHERKONSI STENZ
APPAREIL ET PROCEDE DE MISE A JOUR D'ANTEMEMOIRE/MEMOIRE CENTRALE
PATENT ASSIGNEE:
WANG LABORATORIES, INC., (333566), 600 Technology Park Drive, Billerica, MA 01821, (US), (applicant designated states: BE;DE;FR;GB;NL)
INVENTOR:
PATEL, Bhikoo, J., 23 Broadview Road, Lowell, MA 01852, (US)
LEGAL REPRESENTATIVE:
Behrens, Dieter, Dr.-Ing. (1701), Wuesthoff & Wuesthoff Patent- und Rechtsanwalte Schweigerstrasse 2, 81541 Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 491697 A1 920701 (Basic)
EP 491697 B1 971029
WO 9103785 910321
APPLICATION (CC, No, Date): EP 90907677 900328; WO 90US1641 900328
PRIORITY (CC, No, Date): US 405800 890911
DESIGNATED STATES: BE; DE; FR; GB; NL
INTERNATIONAL PATENT CLASS: **G06F-012/08**
NOTE:
No A-document published by EPO
LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9710W4	1027
CLAIMS B	(German)	9710W4	806
CLAIMS B	(French)	9710W4	1138
SPEC B	(English)	9710W4	3566
Total word count - document A			0
Total word count - document B			6537
Total word count - documents A + B			6537

INTERNATIONAL PATENT CLASS: **G06F-012/08**

...SPECIFICATION the memory system and the main memory of the memory system employ data from a **cache** . To avoid over- **writing** **newly** **written** data in main memory with **old** data from the **cache** a comparison is made of the main memory address of the data to be **written** with the addresses where **data** was **changed** by sources other than the cache. The addresses are stored in a buffer stack. If...

43/3,K/15 (Item 15 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00362142
Supporting long fields of recoverable database records in a computer system.

Unterstutzung langer Felder zuruckzugewinnender Datenbasisaufzeichnungen in einem Rechnersystem.
Prise en charge de longs champs de fiches de base de donnees restaurables dans un systeme a calculateur.
PATENT ASSIGNEE:
International Business Machines Corporation, (200120), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: BE;CH;DE;ES;FR;GB;IT;LI;NL;SE)
INVENTOR:
Camp, Laura Susan, 3608 Cookstown, Austin Texas 78759, (US)
Elliott, Linda Carolyn, 1602 Springer Lane, Austin Texas 78758, (US)
Lehman, Tobin Jon, 175 Newell Avenue, Los Gatos California 95030, (US)
Lindsay, Bruce Gilbert, 1185 Settle Avenue, San Jose California 95025, (US)
LEGAL REPRESENTATIVE:
Bailey, Geoffrey Alan (27921), IBM United Kingdom Limited Intellectual Property Department Hursley Park, Winchester Hampshire SO21 2JN, (GB)
PATENT (CC, No, Kind, Date): EP 336548 A2 891011 (Basic)
EP 336548 A3 910529
APPLICATION (CC, No, Date): EP 89301836 890224;
PRIORITY (CC, No, Date): US 179469 880408
DESIGNATED STATES: BE; CH; DE; ES; FR; GB; IT; LI; NL; SE
INTERNATIONAL PATENT CLASS: **G06F-011/00**
ABSTRACT WORD COUNT: 219

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF1	592
SPEC A	(English)	EPABF1	8878
Total word count - document A			9470
Total word count - document B			0
Total word count - documents A + B			9470

INTERNATIONAL PATENT CLASS: **G06F-011/00**

...SPECIFICATION value, and construct a new long field descriptor;
For each long field contained in the **record** which is to be partially replaced, employ algorithm 8 to **update** the long field value and construct a modified long field descriptor;
Log the **old** and **new** values of the modified long field descriptors (along with **old** and **new** values of any other **modified** fields) and **copy** the LSN of the log **record** to the data file page header in the database **buffer** pool;
For each modified long field, replace its long field descriptor in the record with...

43/3,K/16 (Item 16 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00306062
Digital data processing system.
Digitales Datenverarbeitungssystem.
Systeme du traitement de donnees numeriques.
PATENT ASSIGNEE:
DATA GENERAL CORPORATION, (410940), Route 9, Westboro Massachusetts 01581

, (US), (applicant designated states: AT;BE;CH;DE;FR;GB;IT;LI;LU;NL;SE)
INVENTOR:
Bratt, Richard Glenn, 9 Brook Trail Road, Wayland Massachusetts 01778, (US)
Clancy, Gerald F., 13069 Jaccaranda Center, Saratoga California 95070, (US)
Gavrin, Edward S., Beaver Pond Road RFD 4, Lincoln Massachusetts 01773, (US)
Gruner, Ronald Hans, 112 Dublin Wood Drive, Cary North Carolina 27514, (US)
Mundie, Craig James, 136 Castlewood Drive, Cary North Carolina, (US)
Schleimer, Stephen I., 1208 Ellen Place, Chapel Hill North Carolina 27514 , (US)
Wallach, Steven J., 12436 Green Meadow Lane, Saratoga California 95070, (US)
LEGAL REPRESENTATIVE:
Robson, Aidan John et al (69471), Reddie & Grose 16 Theobalds Road, London WC1X 8PL, (GB)
PATENT (CC, No, Kind, Date): EP 300516 A2 890125 (Basic)
EP 300516 A3 890426
EP 300516 B1 931124
APPLICATION (CC, No, Date): EP 88200921 820521;
PRIORITY (CC, No, Date): US 266413 810522; US 266539 810522; US 266521 810522; US 266415 810522; US 266409 810522; US 266424 810522; US 266421 810522; US 266404 810522; US 266414 810522; US 266532 810522; US 266403 810522; US 266408 810522; US 266401 810522; US 266524 810522
DESIGNATED STATES: AT; BE; CH; DE; FR; GB; IT; LI; LU; NL; SE
RELATED PARENT NUMBER(S) - PN (AN):
EP 67556 (EP 823025960)
INTERNATIONAL PATENT CLASS: **G06F-009/46** ; **G06F-012/14**
ABSTRACT WORD COUNT: 122

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPBBF1	1018
CLAIMS B	(German)	EPBBF1	868
CLAIMS B	(French)	EPBBF1	1115
SPEC B	(English)	EPBBF1	154256
Total word count - document A			0
Total word count - document B			157257
Total word count - documents A + B			157257

INTERNATIONAL PATENT CLASS: **G06F-009/46** ...

... **G06F-012/14**

...SPECIFICATION system.
The system of the invention uses an object based architecture, known per se from **COMPUTER** ARCHITECTURE **NEWS** , October 1980, pages 4-11; J. RATTNER et al.
Prior date processing systems have not...

43/3,K/17 (Item 17 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00306058

Digital data processing system.
Digitales Datenverarbeitungssystem.
Systeme de traitement de donnees numeriques.

PATENT ASSIGNEE:
  DATA GENERAL CORPORATION, (410940), Route 9, Westboro Massachusetts 01581 , (US), (applicant designated states: AT;BE;CH;DE;FR;GB;IT;LI;LU;NL;SE)

INVENTOR:
  Bachman, Brett L., 214 W. Canton Street Suite 4, Boston Massachusetts 02116, (US)
  Bernstein, David H., 41 Bay Colony Drive, Ashland Massachusetts 01721, (US)
  Bratt, Richard Glenn, 9 Brook Trail Road, Wayland Massachusetts 01778, (US)
  Clancy, Gerald F., 13069 Jaccaranda Center, Saratoga California 95070, (US)
  Gavrin, Edward S., Beaver Pond Road RFD 4, Lincoln Massachusetts 01773, (US)
  Gruner, Ronald Hans, 112 Dublin Wood Drive, Cary North Carolina 27514, (US)
  Jones, Thomas M. Jones, 300 Reade Road, Chapel Hill North Carolina 27514, (US)
  Katz, Lawrence H., 10943 S. Forest Ridge Road, Oregon City Oregon 97045, (US)
  Mundie, Craig James, 136 Castlewood Drive, Cary North Carolina, (US)
  Pilat, John F., 1308 Ravenhurst Drive, Raleigh North Carolina 27609, (US)
  Richmond, Michael S., Fearringtn Post Box 51, Pittsboro North Carolina 27312, (US)
  Schleimer Stephen I., 1208 Ellen Place, Chapel Hill North Carolina 27514, (US)
  Wallach, Steven J., 12436 Green Meadow Lane, Saratoga California 95070, (US)
  Wallach, Walter, A., Jr., 1336 Medfield Road, Raleigh North Carolina 27607, (US)

LEGAL REPRESENTATIVE:
  Robson, Aidan John et al (69471), Reddie & Grose 16 Theobalds Road, London WC1X 8PL, (GB)

PATENT (CC, No, Kind, Date): EP 290111 A2 881109 (Basic)
EP 290111 A3 890503
EP 290111 B1 931222

APPLICATION (CC, No, Date): EP 88200917 820521;
PRIORITY (CC, No, Date): US 266404 810522
DESIGNATED STATES: AT; BE; CH; DE; FR; GB; IT; LI; LU; NL; SE
RELATED PARENT NUMBER(S) - PN (AN):
  EP 67556 (EP 823025960)
INTERNATIONAL PATENT CLASS: **G06F-009/30**
ABSTRACT WORD COUNT: 123

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPBBF1	1044
CLAIMS B	(German)	EPBBF1	890
CLAIMS B	(French)	EPBBF1	1185
SPEC B	(English)	EPBBF1	154314
Total word count - document A			0
Total word count - document B			157433
Total word count - documents A + B			157433

INTERNATIONAL PATENT CLASS: **G06F-009/30**

...SPECIFICATION interference by users. Yet another feature is a data processing system having a flexible internal **structure** capable of performing multiple, concurrent operations and comprised of a plurality of separate, independent processors...
...a central communications and memory node. The communications and memory node is also an independent **processor** having separate and independent microinstruction control. The memory processor is internally comprised of a plurality...is utilized as a 32 bit wide general register array. A third portion GRF 506, **LENGRF** 1936, is a 32 bit wide register array for storing length fields of logical descriptors...

...a general register for storing data. Primary data path from MEM 112 to FU 120 **is** through MOD Bus 140, which provides inputs to OFFGRF 1934. As indicated in Fig. **19** , data **may** be **transferred** from OFFGRF **1934** to inputs of AONGRF 1932 and LENGRF 1936 through various interconnections. Similarly, outputs from LENGRF **1936** and AONGRF 1932 may be **transferred** to inputs of AONGRF 1932, **OFFGRF** 1934, and LENGRF 1936. Output of OFFGRF 1934 is connected to inputs of DESP 1910...

43/3,K/18 (Item 18 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00224989
Write request buffering apparatus.
Gerat zur Pufferung von Schreibanforderungen.
Appareil de tamponnage de demandes d'ecriture.
PATENT ASSIGNEE:
MIPS COMPUTER SYSTEMS, INC., (775000), 930 Arques Avenue, Sunnyvale, California 94086-3650, (US), (applicant designated states: AT;BE;CH;DE;ES;FR;GB;GR;IT;LI;NL;SE)
INVENTOR:
Mills, Marvin Anson Jr., PO Box G, Mountain View California 94042, (US)
Crudele, Lester Martin, 82 Flavell Road, Groton Massachusetts 01450, (US)
LEGAL REPRESENTATIVE:
Pilch, Adam John Michael et al (50481), D. YOUNG & CO., 21 New Fetter Lane, London EC4A 1DA, (GB)
PATENT (CC, No, Kind, Date): EP 244540 A2 871111 (Basic)
EP 244540 A3 881026
EP 244540 B1 930915
APPLICATION (CC, No, Date): EP 86310038 861222;
PRIORITY (CC, No, Date): US 860304 860505
DESIGNATED STATES: AT; BE; CH; DE; ES; FR; GB; GR; IT; LI; NL; SE
INTERNATIONAL PATENT CLASS: **G06F-013/16**
ABSTRACT WORD COUNT: 117

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPBBF1	1907
CLAIMS B	(German)	EPBBF1	855
CLAIMS B	(French)	EPBBF1	1099
SPEC B	(English)	EPBBF1	5658
Total word count - document A			0
Total word count - document B			9519
Total word count - documents A + B			9519

INTERNATIONAL PATENT CLASS: **G06F-013/16**

...SPECIFICATION the subsystem into slices, each slice comprising storage for 8 of the 32 address bits **in** each of four buffer ranks, 9 of the 36 data bits in each of the...

...and 2b, the write buffer subsystem 10 shown in Fig. 1 will be described in **greater** detail. It will be seen that the write **buffer** 10 comprises four **buffer** ranks, each comprising a data rank 70, a word address rank 80, and a "valid rank" 50. Each of **the** valid ranks 50a, 50b, 50c and 50d holds four bits of information, each **bit** indicating whether a corresponding byte in **the** corresponding data rank is valid. **The valid** bits are set in accordance with the **control** signals from the CPU 12 **which** accompany any write request. The outputs of the valid ranks 50 are fed back to the inputs through OR gates 52a, 52b, 52c and 52d as hereinafter **described** . The valid ranks 50 are **clocked** by an ungated clock signal, and as long as the second inputs of OR gates...

...The control signals from the CPU 12 include an ACCESSTYPEIN signal, which indicates whether the **write** request is for **writing a** byte, half word, tribyte or

43/3,K/19 (Item 19 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00217674
System and method for controlling network bus communications for tightly coupled information among distributed programmable controllers.
System und Verfahren zur Busubertragungssteuerung fur eng gekoppelte Nachrichten zwischen verteilten programmierbaren Steuergeraten.
Systeme et methode pour commander les communications par bus d'informations a couplage rigide entre des appareils de commande programmables distribues.
PATENT ASSIGNEE:
SIEMENS AKTIENGESELLSCHAFT, (200520), Wittelsbacherplatz 2, D-80312 Munchen, (DE), (applicant designated states: DE;FR;GB;IT;NL;SE)
INVENTOR:
Fulton, Temple L., 1508 Stateline Road, Elizabethton TN 37643, (US)
Perkins, William O., Route 8, Box 141, Johnson City TN 37601, (US)
LEGAL REPRESENTATIVE:
Abbott, David John et al (27491), Abel & Imray Northumberland House 303-306 High Holborn, London, WC1V 7LH, (GB)
PATENT (CC, No, Kind, Date): EP 200365 A2 861105 (Basic)
EP 200365 A3 890628
EP 200365 B1 930922
APPLICATION (CC, No, Date): EP 86302360 860327;
PRIORITY (CC, No, Date): US 719174 850403
DESIGNATED STATES: DE; FR; GB; IT; NL; SE
INTERNATIONAL PATENT CLASS: **G06F-013/366** ; G05B-019/05
ABSTRACT WORD COUNT: 213

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPBBF1	3528

CLAIMS B (German) EPBBF1 1643
CLAIMS B (French) EPBBF1 2203
SPEC B (English) EPBBF1 9204
Total word count - document A 0
Total word count - document B 16578
Total word count - documents A + B 16578

INTERNATIONAL PATENT CLASS: **G06F-013/366** ...

...SPECIFICATION This allows level 1 code to be started from the OS or accessed by the **timer** interrupt service routine (ISR).
Referring to Fig. 13a, on power-up, each module initializes itself and begins communications with the PC. When the dead bus timeout interval **expires** , the module which checks this will mask **all** interrupts and **call** (false **interrupt**) the startup **supervisor** in the AM unit only. **the** start-up **supervisor** (see Fig. 14) calls the Do Poll module (Fig. 15), which sets up the hardware timer **to** do a delay and stores the vector to which it should branch **at** the end of the delay. Then Do Poll returns to the timeout checker, which restores the interrupt mask and returns **to** the executive. The first false interrupt is complete.
When the delay expires, a normal level...receive. The timer is not programmed by NM/PM units. Note that PMs become AMs **in** the dead bus checker if the AM has stopped communicating. Otherwise there are no false ...

...network to a PC and from a PC to the network are shown in Fig. **21** . Data frames from the network are loaded **in** a Receive **Buffer** until the CRC and length are verified. When the data is determined to be good from a particular node, e.g. N, the pointer to that Receive **Buffer** is swapped for the pointer **to** the **old** **node** N **buffer** . The **New** Data Flag is then set for the new node telling the operating system that new data is available. If the CRC or **length** is not good, then no **buffer** **swap** takes place, the bad data is not used and the same **buffer** is used for the next **frame** received.
When the operating system begins to fill the SFIC 18 for the PC **transfer** , it first checks the New Data Flag for each node. If the flag is set...

...cleared keeping the most recent data in the SFIC chip 18.
When data is read **from** the SFIC RAM the operating system writes the data into one of te two transmit...

43/3,K/20 (Item 20 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00213669
Method to test and set data in a record on disk in one atomic input/output operation.
Verfahren zum Testen und Setzen von Daten in einen Datensatz auf einer Platte in eine atomaren Ein/Ausgabeoperation.
Methode pour tester et mettre des dossiers dans un enregistrement sur disque en une operation entree/sortie atomique.
PATENT ASSIGNEE:
International Business Machines Corporation, (200120), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: DE;FR;GB;NL)

INVENTOR:
  Wijnen, Hubertus F. A., Schagen 33, NL-3461 GL Linschoten, (NL)
  Pennings, Henricus M. A., Trompenbergerweg 59, NL-1217 BD Hilversum, (NL)
LEGAL REPRESENTATIVE:
  Monig, Anton, Dipl.-Ing. (8591), IBM Deutschland Informationssysteme GmbH, Patentwesen und Urheberrecht, D-70548 Stuttgart, (DE)
PATENT (CC, No, Kind, Date): EP 240616 A1 871014 (Basic)
                              EP 240616 B1 930324
APPLICATION (CC, No, Date):   EP 86200577 860404;
PRIORITY (CC, No, Date): EP 86200577 860404
DESIGNATED STATES: DE; FR; GB; NL
INTERNATIONAL PATENT CLASS: **G06F-013/16 ; G06F-003/06**
ABSTRACT WORD COUNT: 113

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPBBF1	2318
CLAIMS B	(German)	EPBBF1	1245
CLAIMS B	(French)	EPBBF1	1613
SPEC B	(English)	EPBBF1	5643
Total word count - document A			0
Total word count - document B			10819
Total word count - documents A + B			10819

INTERNATIONAL PATENT CLASS: **G06F-013/16** ...

... **G06F-003/06**

...SPECIFICATION on a disk or DASD (Direct Access Storage Device) while maintaining the rest of the **record** unaltered. The first I/O operation serves to transfer the entire record from disk to main memory, whereupon the data in memory is changed under control of a program running **in** the CPU, and subsequently the **updated** record is written **back** to the DASD in a second I/O operation, the **new** version overlaying the **original** .

In some instances it is desirable or even necessary to set or **change data** in a **record** registered on DASD under the guarantee that no other access takes place **while the update** is **being** made. Under such circumstances the above mentioned change in two successive I/O operations **cannot** be used, because other conflicting accesses might occur in between.

An example of the above...

43/3,K/21   (Item 21 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00163682
Diskette drive and media type determination mechanism.
Vorrichtung zur Typenbestimmung eines Diskettenlaufwerks und -Formats.
Dispositif pour determiner le type d'unite de disque souple et de format.
PATENT ASSIGNEE:
  International Business Machines Corporation, (200120), Old Orchard Road, Armonk, N.Y. 10504, (US), (applicant designated states: BE;CH;DE;FR;GB;IT;LI;NL;SE)
INVENTOR:

Berens, Robert Eugene, 660 NE 32nd Street, Boca Raton, FL 33431, (US)
Bradley, David John, 20839 Sonrisa Way, Boca Raton, FL 33433, (US)
Buckley, Linda Kay, 4403 Ganymede Drive, Austin, TX 78727, (US)
Dayan, Richard Alan, 22244 Boca Rancho Drive Apt. C., Boca Raton, FL 33428, (US)
Smith, Bruce Alan, 5287 Greenwood Drive, Delray Beach, FL 33445, (US)
LEGAL REPRESENTATIVE:
Lattard, Nicole (16571), Compagnie IBM France Departement de Propriete Intellectuelle, F-06610 La Gaude, (FR)
PATENT (CC, No, Kind, Date): EP 167826 A2 860115 (Basic)
EP 167826 A3 870819
EP 167826 B1 910911
APPLICATION (CC, No, Date): EP 85106932 850605;
PRIORITY (CC, No, Date): US 630608 840713
DESIGNATED STATES: BE; CH; DE; FR; GB; IT; LI; NL; SE
INTERNATIONAL PATENT CLASS: G11B-023/40; G11B-019/02; **G06F-013/42** ; G11B-020/12
ABSTRACT WORD COUNT: 64

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	EPBBF1	786
CLAIMS B	(German)	EPBBF1	738
CLAIMS B	(French)	EPBBF1	857
SPEC B	(English)	EPBBF1	7018
Total word count - document A			0
Total word count - document B			9399
Total word count - documents A + B			9399

...INTERNATIONAL PATENT CLASS: **G06F-013/42**

...SPECIFICATION affect compatibility.
Of course, a possible solution is for the user to connect both an **old** style **disk** and a **new** style **disk** to his **computer** . The **old** style **disk** can be used for reading the previously generated **disk** and possibly **writing** on disks to be transported to other users who are still confined to the old...

43/3,K/22 (Item 22 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00459165 **Image available**
UNIVERSAL EPISTEMOLOGICAL MACHINE (A.K.A. ANDROID)
MACHINE EPISTEMOLOGIQUE UNIVERSELLE (ANDROIDE A.K.A.)
Patent Applicant/Assignee:
DATIG William E,
Inventor(s):
DATIG William E,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9849629 A1 19981105
Application: WO 98US8527 19980427 (PCT/WO US9808527)
Priority Application: US 97847230 19970501; US 97876378 19970616; US 9833676 19980303
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)

AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES FI GB GE GH GM GW HU ID IL IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG US UZ VN YU ZW GH GM KE LS MW SD SZ UG ZW AM AZ BY KG KZ MD RU TJ TM AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN ML MR NE SN TD TG
Publication Language: English
Fulltext Word Count: 265553

Main International Patent Class: **G06F-015/18**
Fulltext Availability:
Claims

Claim
... In other sentence constructions, such as those found in the works of the more innovative **writers** , this subject-predicate structure is often altered 1 5 intentionally. To the extent that an...same ultimately real form-the soul. The phenomenological causations of the universe, alono with the **remaining** three C's and the arbitrary forms of existence, place verb tense and mathematical functions...

...the hyphen in the expression English-speaking androids. Neither do they account for paragraph structure, **writing** style, and the ordinary conversational use of language. The remainder of English grammar thus attempts...

43/3,K/23 (Item 23 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00431190 **Image available**
LOG BASED DATA ARCHITECTURE FOR A TRANSACTIONAL MESSAGE QUEUING SYSTEM
ARCHITECTURE DE DONNEES A JOURNALISATION POUR SYSTEME TRANSACTIONNEL DE GESTION DE FILES D'ATTENTE DE MESSAGES
Patent Applicant/Assignee:
MITSUBISHI ELECTRIC INFORMATION TECHNOLOGY CENTER AMERICA INC,
Inventor(s):
WONG David W H,
SCHWENKE Derek L,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9821654 A1 19980522
Application: WO 97US20561 19971111 (PCT/WO US9720561)
Priority Application: US 9630905 19961114
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AU CA CN IL JP KR MX NO NZ SG AT BE CH DE DK ES FI FR GB GR IE IT LU MC NL PT SE
Publication Language: English
Fulltext Word Count: 104413
Main International Patent Class: **G06F-011/00**
International Patent Class: **G06F-12:00**
Fulltext Availability:
Detailed Description

Detailed Description
... magnitude less time than previous systems,
while at the same time establishing an efficient forward **writing**

mechanism to prevent the need for searching through unordered sectors.

In one embodiment, a circular wrap around **buffering** system is used in which a modification of a **previous** sector is made by appending a **new** **record** at the last sector to indicate that the state of a **file** has **changed** , thus to reuse previous blocks that have been freed and no longer hold valid messages...

43/3,K/24 (Item 24 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00403991 **Image available**
REDUNDANT DISC COMPUTER HAVING TARGETED DATA BROADCAST
ORDINATEUR A DISQUES REDONDANTS AVEC DIFFUSION DE DONNEES CIBLEES
Patent Applicant/Assignee:
STORAGE COMPUTER CORPORATION,
Inventor(s):
SARKOZY Andras,
VALENTINO James,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9744735 A1 19971127
Application: WO 97US8253 19970515 (PCT/WO US9708253)
Priority Application: US 96652636 19960521
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AU CA CN JP KR AT BE CH DE DK ES FI FR GB GR IE IT LU MC NL PT SE
Publication Language: English
Fulltext Word Count: 9921

Main International Patent Class: **G06F-011/34**
International Patent Class: **G06F-11:00** ...
... **G06F-11:08** ...

... **G06F-11:14** ...

... **G06F-11:16** ...

... **G06F-11:22**

English Abstract
A redundant **disk** computer system (100) providing targeted data broadcast on the data bus (72) to a plurality...

...data bus (72), such as computer central memory (60) and a plurality of storage media **disks** (parity (50) and data (55)) under control of a real time operating system. The redundant array **computer** operating system (100) provides the **control** and selected designation of the **disk** adapters (95) as targeted receivers to read data "broadcast" over the data bus, providing simultaneous **transfer** of data over the data bus (72). Each enhanced **disk** adapter further includes exclusive-OR logic (93A) thereon to provide direct calculation of parity from the **newly** received data and a subsequently received **old** data on a single subsequent data bus cycle.

43/3,K/25 (Item 25 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00376001
METHOD AND APPARATUS FOR RELIABLE DISK FENCING IN A MULTICOMPUTER SYSTEM
PROCEDE ET APPAREIL POUR LA DELIMITATION FIABLE DE DISQUES DANS UN SYSTEME MULTIORDINATEUR
Patent Applicant/Assignee:
SUN MICROSYSTEMS INC,
Inventor(s):
MATENA Vladimir,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9716744 A1 19970509
Application: WO 96US17603 19961104 (PCT/WO US9617603)
Priority Application: US 95552316 19951102
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AT BE CH DE DK ES FI FR GB GR IE IT LU MC NL PT SE
Publication Language: English
Fulltext Word Count: 7837

International Patent Class: **G06F-11:00**
Fulltext Availability:
Detailed Description

Detailed Description
... g. a failed node), it is not allowed to change its NK. If such a **node** attempts to read or **write** to a **disk** , the controller finds a mismatch
between the **new** CK value and the **old** NK value

When a **node** is started, its NK is initialized to a 0 value

Procedures for Calcularine Values of...

43/3,K/26 (Item 26 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00329718 **Image available**
INDEXING AND MULTIPLEXING OF INTERLEAVED CACHE MEMORY ARRAYS
INDEXATION ET MULTIPLEXAGE DE MATRICES ENTRELACEES DE MEMOIRES D'ATTENTE
Patent Applicant/Assignee:
SILICON GRAPHICS INC,
Inventor(s):
YEAGER Kenneth C,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9612229 A1 19960425
Application: WO 95US13241 19951013 (PCT/WO US9513241)
Priority Application: US 94324124 19941014
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
JP AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE

Publication Language: English
Fulltext Word Count: 14512

Main International Patent Class: **G06F-012/06**
International Patent Class: **G06F-12:10** ...
Fulltext Availability:
Detailed Description

Detailed Description
... a tag while it is in the "refill" state.

Thus, the external interface can simply **write** a block's new state at the end of each refill operation. It does not need to do an atomic operation here. State **changes** for the **Data** **Cache** Tags are listed in Table VI.

Table VI Data **Cache** Refill Way
Old **Old** **New** **New** **Update** Description
Oper State Mod Condition State Mod Cycle
CPU (input) (input) (inputs) (dec) (deq) (CPU...

43/3,K/27 (Item 27 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00323195 **Image available**
METHOD FOR TRANSFER OF DATA FILES FROM A MASS STORAGE DEVICE TO A POST-PROCESSING SYSTEM
PROCEDE DE TRANSFERT DE FICHIERS DE DONNEES D'UNE MEMOIRE DE GRANDE CAPACITE A UN SYSTEME DE POST-TRAITEMENT
Patent Applicant/Assignee:
NOKIA TELECOMMUNICATIONS OY,
JARVENPAA Anssi,
Inventor(s):
JARVENPAA Anssi,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9605703 A2 19960222
Application: WO 95FI417 19950807 (PCT/WO FI9500417)
Priority Application: FI 943668 19940808
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AM AT AU BB BG BR BY CA CH CN CZ DE DK EE ES FI GB GE HU IS JP KE KG KP KR KZ LK LR LT LU LV MD MG MN MW MX NO NZ PL PT RO RU SD SE SG SI SK TJ TM TT UA UG US UZ VN KE MW SD SZ UG AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN ML MR NE SN TD TG
Publication Language: English
Fulltext Word Count: 2705

International Patent Class: **G06F-13:00** ...
Fulltext Availability:
Detailed Description

Detailed Description
... communication device DX or the peripheral device interface OMC can infer which files have been **transferred** to the operating system, which files are full of data, and which

file is currently open. The next data file to be opened is the oldest of the **transferred** files and thus the **new** data are **written** over the **previous** data of the file which was opened. In this way, the **disk** space which the system uses for **storing** data **remains** reasonable.

The post-processing system advantageously utilizes a polling cycle during which it reads (arrow...

43/3,K/28 (Item 28 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00300850 **Image available**
UPDATE MECHANISM FOR COMPUTER STORAGE CONTAINER MANAGER
MOYEN DE MISE A JOUR POUR MODULE DE GESTION D'ELEMENTS DE STOCKAGE D'ORDINATEURS
Patent Applicant/Assignee:
APPLE COMPUTER INC,
Inventor(s):
HARRIS Jared M,
RUBEN Ira L,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9519001 A1 19950713
Application: WO 95US196 19950104 (PCT/WO US9500196)
Priority Application: US 94177853 19940105
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AM AT AU BB BG BR BY CA CH CN CZ DE DK EE ES FI GB GE HU JP KE KG KP KR KZ LK LR LT LU LV MD MG MN MW MX NL NO NZ PL PT RO RU SD SE SI SK TJ TT UA UZ VN KE MW SD SZ AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN ML MR NE SN TD TG
Publication Language: English
Fulltext Word Count: 119635

Main International Patent Class: **G06F-009/44**
Fulltext Availability:
Claims

Claim
... 1, which represents the container itself.
Table of Contents Format. The TOC as represented in **persistent** **storage** consists of a sequence of entries. Each entry corresponds to a single segment of a...completeness, however, Fig. 3 illustrates a typical hardware computer system platform on which the Container **Manager** might run.
The **computer** system of Fig. 3 comprises a CPU 302, main memory 304,, which may be volatile including nonvolatile storage devices, such as a **disk** 312, In typical operation, an application program, together with at least those Container Manager routines which are used by the application program, are retrieved from the **disk** 312 into main memory 304 for execution by the CPU 302,
All of the data...414 control block

416.
The table of contents (TOC) 414 is the set of related **data** structures that organize objects by object IDs, The requirement that objects be kept in sorted...entries.
CMTOCNewValuesTOCGlobalName/CMTOCValueTypeGlobalName - new target values
Defines that portion of an updating TOC that contains **new** values for " **old** " properties and **new** values for **new** properties. This is used to distinguish the "private" TOC of an updating container from objects...

...details.
CMTOCTargetGlobalName/CMTargetContainerName - "pointer" to target container
The "pointing" value to the target container when **updating** . For an appended target, the type is always CMTargetContainerName and the value's offset/size...TOCObject *nextTypeProperty; /* chain of next type/property by increasing ID*/
struct TOCObject *prevTypeProperty; /* chain of **previous** type/property by decr. ID
unsigned short objectF(ags; /* info flags about the object
CMRefCon...

...each of the properties 502, 504 and 542 is defined as follows:
0 1992 Apple **Computer** , Inc.
struct TOCProperty (/* Layout of a TOC object property:
3 0 ListLinks propertyLinks; /* Links to...filename, 0, some size number);
obtain data from file designated by file name in filename **buffer** , using caller's offset and size;
write the data into caller's destination **buffer** ;
return
As with the previous read handlers, this read handler begins by obtaining its base...

...handler then calls CMReadValueDataO using the base value, in order to obtain it in a **buffer** "filename". information which the read handler will interpret as a file name. The read handler...
...filename, obtains the data from the file, and places the result in the callerFs destination **buffer** .
It can be seen that CMReadValueData has now been called recursively yet a fourth time...

...CMReadValueDataO to obtain a file name.
After placing the result in the caller's destination **buffer** , the ...are deleted.
B. Sample Value Handlers
Appendix A of the related DYNAMIC VALUE MECHANISM FOR **COMPUTER** STORAGE CONTAINER **MANAGER** patent application is a header file for a sample set of value handlers for a...to collectively as the "OPT address".
The OPTs will work their way into the value **updating** instructions generated at close time for use in

addressing the original values at open time...list, These are the updating instructions that open-time processing will need to bring the **old** objects to the state they have at the time of this close. The instructions are...CMReadValueDatao and CMWriteValueDatao, The handlers are defined only to move data to and from a **buffer** location (not by accident). Thus the code **buffers** the updating information and calls CMReadValueDatao or CMWriteValueDatao only when needed, The **buffer** size is defined in a header file as UpdateBufSize.
Note that this same **buffered** 1/0 is supported for TOC 1/0 as-well, In that case the I...

...through the I/O handlers instead of API calls to CMReadValueDatao and CMWriteValueDatao, but the **buffering** algorithms are the same. The moving of data between the **buffer** and the container (TOC fields in this case) utilize the same handlers described above. Unlike updating **buffering** , TOC **buffering** is optional, This is because **buffering** may be performed by the I/O handlers. A header file controls this by defining the input and output TOC **buffer** sizes (TOCInputBufSize and TOCOutputBufSize, respectively). Defining either or both of these as 0 turns off the corresponding TOC **buffering** through routines supplied with the Container Manager,
G, Open-time Processing
The following lists the...have parameters that can be directly used in many of the corresponding API calls, If **update** **recording** wasn't suppressed, a loop condition would result, with objects being added to the touched...

...processed,
(7)o Apply object and property deletions,
After the TOC is loaded and value **updates** applied, the remaining delete object and property **updating** instructions can be processed using the special TOC #1 property of the container's private TOC.
At this point the target And the **updating** container have been opened, The **updater** 's CCB pointer is returned to the user as the container refNum, The diagram of...file for a set of basic container handlers, and their metahandler, used by the Container **Manager** when doing update operations on a target container. Appendix G sets forth the handlers themselves...

...aspects of the TOC in-memory which are needed for a particular operation after an **update** container is opened. The embodiments described herein were chosen and described in order to best...

...the following claims and their equivalents.
APPENDIX A
```
#define DebugEmbedding 0
/* -----------
```

<<< CMContainerops.c >>>
Container **Manager** Container operations
Ira L. Ruben
12/02/91
Copyright Apple Computer, Inc. 1991-1992...i.e., a container within a container. Containers may also be 1lappeneded" or descrete fo **updating** other containers.

#incLude <stddef.h>
#incLude <string.h>
#incLude <stdio.h>
...

...I oba L Namejh-11
#endif
#ifndef CONTAINEROPS.
#inc:Lude@IoContainers.hll
#endif
#ifndef **BUFFEREDIO**
#i nc L ude@@" **Bufferedl** 07h
#endif
#ifndef **UPDATING**
#i nc: Lude@@" **Updat** i ni7hll
#end i f
#ifndef HANDLERS
i nc: I udj 71 H and...Q, i3, i4, 5);
return (faLsj);

*majorVersion = majorVer;
*minorVersion = minorVer;
/* Set the TOC **buffer** size to be used to read this container...

if (majorVer > 1)
*tocBufSize = ((unsigned...

...here indicates failure. The missing parameter (carfut how you read that) points to a string **buffer** pointer or a pointer to NULL. It indicates whether the handier for the operationType is...

...0 on the first occurrence of a missing handler. The caller has to own the **buffer** . Hence the extra indirection. After the entire vector is build, the caller checks for any if we couldn't allocate the string **buffer** , and (3) non-nuLl (and not 0xFFFFFFFF) indicating we had a previous failure and we allocated the **buffer** . The reason we use a dynamic **buffer** at all is that the insert can get potentially large if for some...

...this is the first missing routine... if Wmissing = (char *)CMmatioc(1024)) == NULOW allocate the insert **buffer** */
ERROR1(CM err-NoMissingBuffer, TYPENAME); /* huh?
return (t7rue), /* failure will propagate
else
**missing...

...the standard API "CM... interfaces. Although these objects are truely in the TOC, we never **write** them out to the container. So reregistering them is necessary every time we open a...control block. For input, the

container is opened, the Label read and TOC Loaded. For **writing** the container is simply opened. No matter what the mode, all the other container control...do the cleanup:*/
#define UndoOpeno CMfcLose(container); /* close the container cmFreeAtIlOBuffers(container); /* free 1/0 **buffers** *A cmFreeALtGLobaLNames(&container->giobatNameTabte); /* frre globats*/ cmFreeTOC(container, &container->toc); /* free TOC data*/
2 5...5 return (NULL);
/* Set up a object TOC for this container. There are times during **updating** target /* containers that the TOC we're using is not for the current container, but...

...3 5 /* be fitted in shortly as a function of whether we're reading or **writing** . The flags,*/
/* and generation are initially 0 for reading (i.e., when called by /* CMOpenContainero), and caller specified values when **writing** (i.e., when called by*/ /* CMOpenNewContainero). In the read case the actual values get fitted...container->deLetedVaLues);
cmlnitList(&container->embeddedContainers);
cmlnitList(&container->activelOBuffers);
15 /* Determine whether we're reading or **writing** . If **writing** , there is not too much to do*/ /* since there isn't anything yet created. For...
...mode the open mode is llrb+ll. The intent is to open a container for **updating** , i.e., reading and **writing** , but preserve the*/ current contents of the file. **writing** The mode here is llwb+ll. The intent is to create the container if*/ 3...

...exist, set its file size to 0 (trucnate it), and*/
to allow BOTH reading and **writing** (**update**). The API allows reading*/ of stuff previously **written** (why not?). reading The read open mode is llrbll (no plus). An existing container is to
3 5 be opened for input (reading) only. it cannot be **written** to for*/ **updating** . **updating** The read open mode is Ilrb+11. An existing container is to be opened*/
0 for **updating** . This is also used for reusing free space. */
/* The 'lb" in these modes is just...

...also checked by /* CMOpen[NewiEontainero. The kCMWriting flag is NOT set by CMOpenNewContainero for /* the **updating** cases. That tells us which is open case we got here so we know to...

...6 0
if (useFLags & kCMConverting) /* if converting... container->refCon = CMfopen(container, attributes.Ilrb+"); /* ... open for **update** ... container->...offset + 1*/
container->useFtags = (unsigned short)(useFlags 1= kCMWriting);
> else if (useFLags & kCMWriting) (/* if just **writing** ... container->refCon = CMfopen(container, attributes,"wb+"); /* ... open **update** & trunc else /* if reading... if ((useFlags & kMeuseFreeSpace) != 0) f /* if reuse free space... container->refCon = CMfopen(container,attrikxjtes.Ilrb+");/* ... open for **updating** */ container->useFLags = (unsigned short)(useFiags 1= kCMWriting);
5 else
container->refCon = CMfopen(container, attributes.Ilrbl...also tells it what 4 is. The non-private TOC (5 and 6) can be /* **written** as a single walk of the TOC. Since old objects always have IDs < min*/ 6...

...don't care, but I'm telling you anyway!

if (!aborting && container->useFtags & kCMWriting) (/* if **writing**
updates ...
InitRackPatches(&thePatches); /* ... prepare for patching TOC*/
if (!container->usingTargetGLobaLs 11
cmWriteALLGLobatNames(container->privateGLobats))
if (cmWriteALLGlobaiNames(container->gtobaLNameTabLe))
if (cmGenerateUpdates(container)) /* **updates** generated here*/
if (cmWriteTOC(container, container->privateTOC, ALLOBJECTS, MAXUSERID,
&thePatches, &pvtTOCStart, &pvtTOCSize)) (
if (cmWriteTOC(container...

...0) /* ... if flush handler provided*/
ERROR1(CM.err.FtushErr, CONTAINERNAME); /* ... report error if it
occurred*/
/* **writing** */
/* At this point the **updating** container has been fully **written** ,
including its Label.*/ /* ALL that's Left to do is physically close the
container layers...updatingContainer = container; /* work back to this
container*/
container = container->targetcontainer; /* ... starting at the target*/
cmFreeALLIC) **Buffers** (container); /* free 1/0 **buffers** separately*/
for (;;) (/* loop up Layers*/
if (container->pointingValue) /* release pointing*/
CMReieaseValue(container->pointingVaLue...

...parentContainer; /* point at parent*/
/* next Layer up*/
return; /* we're now fully closed!*/
/* **updating**
/* From here on, we have a "normal" close... /* Associated with each
container is...

...embeddedVaLue*/ /* in our container control block. /* If we opened the
container for output, then we **write** out the global names to the /*
end of the container. That will add the global names container offsets to
the TOC.*/ /* Then we **write** the TOC itself following the global names.
Lastly, if we're not an*/ /* embedded container...

...in when we call cMWriteTOCO below. Actually the correct value is back /*
patched into the **written** container.
if (laborting && container->useFLags & kCMWriting)
InitBackPatches(&thePatches);
if (cmWriteALLGLobatNames(container->gLobaLWameTabLe))
if...

...embeddedVaLue == NULL) /* if not embedded...
if (container->hardLer.cmffLush 1= NULL) /* ... flush all 1/0 **buffers** */
if (CMffLush(container) 1= 0) /* ... if flush handler provided*/
ERROR1(CM.err.FLushErr, CONTAINERNAME); /* ...report error if it
occurred*/
/* **writing**
/* ALL that's left to do is close the container. For an embedded
container we...DynamicVaLuesObject
/* ....
CMAbortContainer - abort use of a container
---The container is closed

WITHOUT further **writing** to the container, i.e., as if it were opened for reading even when opened for **writing** . This is intended to be used to abort processing of the container from unrecoverable errors...

...CMCLoseContainero but with the session abort switch set. That switch causes CMCloseContainero to avoid aLL **writes** but atL the data structure freeing still happens as usual. That way, atL the freeing...CMContainer container)
return ((container MULL) ? NULL : (CMSession)SESSION);
CM-END-CFUNCTIONS
APPENDIX B
/*

-
<<< **Update** .h >>>
Container Manager Container **Updating** Interfaces
Ira L. Ruben
6/16/92

....
.. This file contains the interfaces to the "incremental **update** " processing routines when one **updating** container **updates** its target container. The routines defined here are responsible for maintaining the "touched chain" and...

...data structures Layered on to the main data structures (see TOCEntries.rhc]). See **Update** .c for complete details.
#ifndef **UPDATING**
#define @LJPDATIN6@
#ifndef CMTYPES
#incLud@-11CMTypes7h11
#endif
#ifndef CM API TYPES
#include@@"OaPIT...

...struct TOCProperty;
struct TOCVaLueHdr;
CM CFUNCTIONS
/* The touched List is created for every **updated** object. The head of this List is in /* the touched TOCObject. An entry is created...

...value (header) is */ /* touched. At that time the original object, property, and type IDs are **recorded** .*/
struct TouchedListEntry (/* Layout of a touched List entry:*/
ListLinks touchedListLinks; /* Links to next...The TOCObject pointed to by theObject is "touched". i.e., it is put on the **updating** container's singly Linked chain of touched objects, the touchedChain. The container is passed because...

...TOC, calls this routine to add objects to the touched chain that have the special " **updates** " property. Such objects have associated **updating** instructions (generated by generateVaLueLJpdateso at close time). These must be chained together until the TOC is fully read in. This is because "inserted" (i.e., move) **updates** could refer to objects not yet read in. The touched chain is a convenient pLacel...

...3 0 it is "fed" to cmApptyUpdateso who walks the chain to apply all the **updates** . The chain is then immediately cleared ready to receive new **updates** in the "normal" way. Note, the object is placed on the touched chain only once...

43/3,K/29 (Item 29 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00281629
WRITE ANYWHERE FILE-SYSTEM LAYOUT
DISPOSITION D'UN SYSTEME DE FICHIERS A ECRITURE DANS UNE ZONE NON PREDETERMINEE
Patent Applicant/Assignee:
NETWORK APPLIANCE CORPORATION,
Inventor(s):
HITZ David,
MALCOM Michael,
LAU James,
RAKITZIS Byron,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9429807 A1 19941222
Application: WO 94US6320 19940602 (PCT/WO US9406320)
Priority Application: US 93643 19930603
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
JP AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE
Publication Language: English
Fulltext Word Count: 15104
Main International Patent Class: **G06F-015/40**
Fulltext Availability:
Detailed Description

Detailed Description
... 2326.

As shown in Figure 17L, in a consistency point, the active file system is **updated** by copying the inode of the inode file 2346 into fsinfo block 2302.

However, the blocks 2314, 2320, 2324, and 2306 of the previous consistency point **remain** on **disk** . These blocks are never overwritten when **updating** the file system to ensure that both the **old** consistency point 1830 and the **new**
consistency point 1831 exist on **disk** in Figure 20 during step 540
SNAPSHOTS
The WAFL system supports snapshots. A snapshot is...

43/3,K/30 (Item 30 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00234265 **Image available**
SYSTEM FOR DIVIDING PROCESSING TASKS INTO SIGNAL PROCESSOR AND DECISION-MAKING MICROPROCESSOR INTERFACING

SYSTEME DE SEPARATION DES TACHES DE TRAITEMENT EN TACHES POUR INTERFACAGE AVEC UN PROCESSEUR DE SIGNAUX ET UN MICROPROCESSEUR DE PRISE DE DECISION

Patent Applicant/Assignee:
  STAR SEMICONDUCTOR CORPORATION,
Inventor(s):
  ROBINSON Jeffrey I,
  ROUSE Keith,
  KRASSOWSKI Andrew J,
  MONTLICK Terry F,
Patent and Priority Information (Country, Number, Date):
  Patent: WO 9308524 A1 19930429
  Application: WO 92US8954 19921014 (PCT/WO US9208954)
  Priority Application: US 91776161 19911015
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
  AU CA JP KR AT BE CH DE DK ES FR GB GR IE IT LU MC NL SE
Publication Language: English
Fulltext Word Count: 219172

Main International Patent Class: **G06F-009/00**
International Patent Class: **G06F-09:40**
Fulltext Availability:
  Claims

Claim
... the fifth clock cycle, the host is given preferred access to either read from or **write** to the program RAM. If the host does not need to read or **write** to the program **RAM** , the access port is given access. Alternatively, the host and access ports can be given...

...EPROM 170,, the program RAM bus 155 must be used by the GSP 400a for **writing** to the program RAM 150 (via data RAM bus 125 and switch 194). Thus, a...parallel load shift register 750, and controlled multiplexers 760 and 770. The data to be **written** from the SPROC via the output port 700b is received by the **buffer** 740 from **buffer** 694 of the DFM 600. The twenty-four bits received are then loaded in parallel...

43/3,K/31 (Item 31 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00221691 **Image available**
FAULT TOLERANT NETWORK FILE SYSTEM
SYSTEME DE FICHIER POUR RESEAU INSENSIBLE AUX DEFAILLANCES
Patent Applicant/Assignee:
  EASTMAN KODAK COMPANY,
Inventor(s):
  VINTHER Gordon,
  McGRATH James W,
Patent and Priority Information (Country, Number, Date):
  Patent: WO 9218931 A1 19921029
  Application: WO 92US3001 19920414 (PCT/WO US9203001)
  Priority Application: US 9166 19910423
Designated States:
(Protection type is "patent" unless otherwise stated - for applications

prior to 2004)
AT BE CH DE DK ES FR GB GR IT JP LU MC NL SE
Publication Language: English
Fulltext Word Count: 8259

Main International Patent Class: **G06F-011/20**
International Patent Class: **G06F-11:14**
Fulltext Availability:
Detailed Description

Detailed Description
... memory 31 using the data
SUBSTITUTE SHEET
NVO 92/18931 PCr/US92/03001
22
block **transfer** procedure described above. (Step 442).

After completing this **transfer**, it deallocates the CPU memory block which contains the journal bit map (Step 444) and...

...Referring to Figs. 8(a) and 8(b), the following describes the operation of the **primary** file **server** 15 in returning to operation from a failure. The primary first sends an walivew message the backup fileserver responds by sending the **disk** data block allocation bit map,, the primary replaces its **old** data block allocation bit map with the **newly** arrived allocation bit map, (Step 512), It then proceeds to read each arriving **modified data** block from the dual ported interface and to **write** the block to its disk 26. (Step 514). When the backup fileserver sends a Journal...

43/3,K/32 (Item 32 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00215265
FLUID TRANSFER DEVICE AND METHOD OF USE
DISPOSITIF DE TRANSFERT FLUIDE ET PROCEDE DE MISE EN OEUVRE
Patent Applicant/Assignee:
ARRAY TECHNOLOGY CORPORATION,
Inventor(s):
BRANT William A,
STALLMO David C,
WALKER Mark,
LUI Albert,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9212482 A1 19920723
Application: WO 92US59 19920103 (PCT/WO US9200059)
Priority Application: US 91167 19910104
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
JP
Publication Language: English
Fulltext Word Count: 7529

Main International Patent Class: **G06F-011/10**
Fulltext Availability:
  Detailed Description

Detailed Description
... location in the array is still a "pending blocw' (a data block that has been **Written** to the copyback **cache** storage unit CC but not **transferred** to the array S1 -S5), the directory location pointer for the **data** block is **changed** to point to the " **new** " version rather than to the " **old** " version. The old version is thereafter ignored, and may be written over in subsequent operations...at the same location in the array is still a "pending block" in the controller **buffer** , the directory location pointers for the **data** block are **changed** to point to the " **new** " version rather than to the " **old** " version both in the copyback **cache** storage unit CC and in the **buffer** . The old version is thereafter ignored, and may be **written** over in subsequent operations.

After a **Write** request is processed in this fashion, the contreller 3 immediately sends an acknowledgement to the...

43/3,K/33 (Item 33 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00196051
DATA STORAGE APPARATUS AND METHOD
PROCEDE ET APPAREIL DE MEMORISATION DE DONNEES
Patent Applicant/Assignee:
  SF2 CORPORATION,
Inventor(s):
  JAFFE David H,
  POWERS David T,
  GAJJAR Kumar,
  GLIDER Joseph S,
  IDLEMAN Thomas E,
Patent and Priority Information (Country, Number, Date):
  Patent: WO 9113404 A1 19910905
  Application: WO 91US1252 19910227 (PCT/WO US9101252)
  Priority Application: US 90749 19900302
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
  AT AT AU BB BE BF BG BJ BR CA CF CG CH CH CM DE DE DK DK ES ES FI FR GA GB GB GR HU IT JP KP KR LK LU LU MC MG ML MR MW NL NL NO PL RO SD SE SE SN SU TD TG
Publication Language: English
Fulltext Word Count: 10536

Main International Patent Class: **G06F-011/10**
International Patent Class: **G06F-11:08** ...

... **G06F-11:20**
Fulltext Availability:
  Detailed Description

Detailed Description
... for their respective sector

rows. Thus, if data is wr.itten to sector S4 of **disk** D7. then **updated** check data is **written** into sector S4 of **disk** D10. This is accomplished by reading the **old** check data, re@coding it using the **new** data, and **writing** the new check data to the **disk** . This operation is referred to as a read@ **modify** @ **write** . Similarly, if **data** is **written** to sector SI of **disk** D11, then check data is **written** into sector SI of disk D13. Since there is no overlap in this selection of four disks for **writes** , both read-modify@write operations can be performed in parallel.

A distribution of check data...

43/3,K/34 (Item 34 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00196046
DISK ARRAY SYSTEM
SYSTEME DE PILE DE DISQUES
Patent Applicant/Assignee:
SF2 CORPORATION,
Inventor(s):
IDLEMAN Thomas E,
KOONTZ Robert S,
POWERS David T,
JAFFE David H,
HENSON Larry P,
GLIDER Joseph S,
GAJJAR Kumar,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9113399 A1 19910905
Application: WO 91US1276 19910228 (PCT/WO US9101276)
Priority Application: US 90749 19900302; US 90622 19900406; US 90703 19900406; US 90482 19901022
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AT AT AU BB BE BF BG BJ BR CA CF CG CH CH CM DE DE DK DK ES ES FI FR GA GB GB GR HU IT JP KP KR LK LU LU MC MG ML MR MW NL NL NO RO SD SE SE SN SU TD TG
Publication Language: English
Fulltext Word Count: 26256

Main International Patent Class: **G06F-007/22**
International Patent Class: **G06F-11:10**
Fulltext Availability:
Detailed Description

Detailed Description
... remaining hatched sectors contain check data for their respective sector rows. Thus, if data is **written** to sector S4 of **disk** D7, then **updated** check data is **written** into sector S4 of **disk** D10. This is accomplished by reading the **old** check data, re-coding it using the **new** data, and **writing** the new check data to the **disk** . This operation is referred to as a

read- **modify** - **write** . Similarly, if **data** is **written** to sector S1 of **disk** D11, then check data is **written** into sector S1 of disk D13. Since there is no overlap in this selection of four disks for **writes** , both read@modify@write operations can be performed in parallel.

A distribution of check data...

43/3,K/35 (Item 35 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00169543
EFFICIENT METHOD FOR UPDATING PIPELINED, SINGLE PORT Z-BUFFER
PROCEDE EFFICACE DE MISE A JOUR D'UN TAMPON Z A UN SEUL POINT D'ACCES, EN PIPELINE
Patent Applicant/Assignee:
SILICON GRAPHICS INC,
Inventor(s):
HANNAH Marc R,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9002990 A1 19900322
Application: WO 89US3938 19890911 (PCT/WO US8903938)
Priority Application: US 88427 19880912
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AT BE CH DE FR GB IT JP KR LU NL SE
Publication Language: English
Fulltext Word Count: 8716

Main International Patent Class: **G06F-001/04**
International Patent Class: **G06F-03:153**
Fulltext Availability:
Detailed Description
Claims

Detailed Description
The invention relates to a raster scan, **computer** **controlled** video display system for presenting an image to an observer on a video display apparatus, which system includes a Z- **buffer** for storing Z values to enable the **computer** **controlled** video display system to present a 3-dimensional representation of an image to the observer. More specifically, the invention relates to a method for **updating** the Zbuffer with **new** Z values to replace **old** Z values.

Computer **controlled** video display systems which utilize the conventional
raster scan technique in their operation, present a...

...dimensional representation of an image to an observer by, among other things, utilizing a Z- **buffer** which contains the information that indicates whether one object on the screen is in front of or behind another object. That is, the Z- **buffer** contains information for each pixeron the screen to indicate whether the object will be hidden...from the Z-buffer 1 1. The data bus 28 is coupled between the graphics **update** controller 25 and the frame buffer 1 0 to provide pixel values to and

from...

...alternatively, static random access memory (SRAM) may be utilized for either or both of the **buffers** 1 0 and 1 1; such an implementation would reduce the need for and expense...

...depicted in Figure 3 by a detailed flow chart. The method involves a process for **updating** the Z- **buffer** with **new** Z values to replace **old** Z values in order to permit the presentation of a 3-dimensional representation of an...

...method will be provided. The method of the invention is utilized in a raster scan, **computer** **controlled** video display system which includes a Z- **buffer** for storing Z values and a frame buffer for storing the pixel values. The new...

Claim
1 In a raster scan, **computer** **controlled** video display system for presenting an image to an observer, said system having a Z- **buffer** for storing Z values and a frame **buffer** for storing pixel values, a method for **updating** said Z- **buffer**
with **new** Z values to replace **old** Z values comprising:
calculating a **new** pixel value and a new Z value for each pixel location in a plurality of...

...location in a contiguous group of pixel locations having a pass condition, said Zbuffer being **updated** for the contiguous group of pixel locations having a pass condition, which contiguous group ends with the pixel location immediately preceding said current pixel location.
. In a raster scan, **computer** **controlled** video display system for presenting a 3-dimensional representation of an image to an observer on a video display apparatus, said system having a Z- **buffer** for storing Z values and having a frame **buffer** for storing pixel values, a method for **updating** said Z- **buffer** with **new** Z
values to replace **old** Z-values comprising:
receiving a **new** pixel value and a new Z value for each pixel location in a plurality of...

43/3,K/36 (Item 36 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00153060
PARALLEL MACHINE ARCHITECTURE FOR PRODUCTION RULE SYSTEMS
ARCHITECTURE DE MACHINE PARALLELE POUR DES SYSTEMES DE REGLES DE PRODUCTION
Patent Applicant/Assignee:
MARTIN MARIETTA ENERGY SYSTEMS INC,
Inventor(s):
ALLEN John Daniel Jr,
BUTLER Philip Lee,
Patent and Priority Information (Country, Number, Date):
Patent: WO 8809972 A1 19881215
Application: WO 88US1901 19880609 (PCT/WO US8801901)
Priority Application: US 87976 19870609
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)

AT BE CH DE FR GB IT JP LU NL SE
Publication Language: English
Fulltext Word Count: 138162

Main International Patent Class: **G06F-015/18**
Fulltext Availability:
Detailed Description

Detailed Description
... associated with Artificial
Intelligence,
Another object of the invention is to provide a high
speed **computer** architecture capable of executing large and
complex OPS and OPS-like Production Rule programs at...interrupt, which
brings them out of the
halted condition, the host will clear the interrupt
control port.

The rule processors independently process the
results of the RHS actions sent by the...local CPU address connections.

C-DO to C-D15
This is the 16 bit host **data** bus extension
line 22B of Fig, 6, The naming scheme reflects that of
the 68000...is then started by doing any kind of access such as
a dummy read or **write** , When all RPs are started with C
RUN, the host must do a dummy **write** since it is not
possible to read from all RPs at once, For a single RP,
either a dummy read or **write** will be effective, To effect
a halt of CPU 84, the C-HALT line is...

43/3,K/37 (Item 37 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00144531 **Image available**
DATA DECOMPRESSION USING A POLYNOMIAL COMPUTATION ENGINE
DECOMPRESSION DE DONNEES A L'AIDE D'UN CALCULATEUR POLYNOMIAL
Patent Applicant/Assignee:
ALLIED CORPORATION,
Inventor(s):
ZENO John Richard,
Patent and Priority Information (Country, Number, Date):
Patent: WO 8801414 A1 19880225
Application: WO 87US1677 19870716 (PCT/WO US8701677)
Priority Application: US 8665 19860811; US 86411 19860811; US 86412 19860811; US 86413 19860811; US 86548 19860811
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AT BE CH DE FR GB IT JP LU NL SE
Publication Language: English
Fulltext Word Count: 11605

Main International Patent Class: **G06F-015/72**
Fulltext Availability:
Detailed Description

Detailed Description
... the register 159 is effectively
isolated from any changes occuring at the adder inputs. A **WRITE** strobe is then sent to the appropriate **RAM** 141 or 142 using the **WRITE** lines. The **new** iterated addition value is stored over the **old** . During this **WRITE** operation, any **change** in the **data** fed to the adder 147 is isolated by the 2-5 **temporary** **storage** register 159,, eliminating a feedback condition.

The output of counter 152 is monitored within the...

43/3,K/38 (Item 38 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00125575
METHOD OF INSERTING AND REMOVING ISOCHRONOUS DATA INTO A SEQUENCE OF NONISOCHRONOUS DATA CHARACTERS WITHOUT SLOT ALLOCATION ON A COMPUTER NETWORK
PROCEDE POUR INSERER DES DONNEES ISOCHRONES DANS UNE SEQUENCE DE CARACTERES DE DONNEES NON ISOCHRONES ET POUR LES EN RETIRER SANS ATTRIBUTION DE TRANCHE DE TEMPS SUR UN RESEAU INFORMATIQUE
Patent Applicant/Assignee:
BURROUGHS CORPORATION,
Inventor(s):
BELL John Leslie,
ISAMAN David Lee,
OSMAN Fazil Ismet,
Patent and Priority Information (Country, Number, Date):
Patent: WO 8503829 A1 19850829
Application: WO 85US244 19850215 (PCT/WO US8500244)
Priority Application: US 84351 19840217
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
JP
Publication Language: English
Fulltext Word Count: 6958

International Patent Class: **G06F-13:02**
Fulltext Availability:
Detailed Description

Detailed Description
... one of the isochronous character
slots between the S and E character into the data **buffer** 36* For example, station S4 may copy the I1 character at time t12*
Station SS...

...in its shift register 31.

is As an example, Figure 4 shows that station SS **writes** a **new** isochronous data character I1' over the **previous** character I1 at time' t13. , Station SS may also store a copy of the other isochronous data character I2' in its data **buffer** 36

at time t14.
Suitably, the determination of which slot a partic
ular station can **write** into and which slot a particular station can read from is made by designating one of the **stations** as a **master** ; and by sending requests to the master prior to **writing** or receiving data in the isochronous frame. This master then does the slot-allocating book...
?

16/3,K/1 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014845359 **Image available**
WPI Acc No: 2002-666065/200271
Related WPI Acc No: 1999-494629; 2002-339699; 2002-609207; 2002-636953
XRPX Acc No: N02-527002

Disk-write operation co-ordination method for database system, involves assigning higher priorities to data items corresponding to entries reacting latter one of two queues

Patent Assignee: ORACLE CORP (ORAC-N)
Inventor: **BAMFORD R J ; BRIDGE W H ; BROWER D ; CHAN W W S ; CHANDRASEKARAN S ; MACNAUGHTON N ; SRIHARI V**
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20020099729	A1	20020725	US 98199120	A	19981124	200271	B
			US 2001274270	P	20010307		
			US 200292047	A	20020304		

Priority Applications (No Type Date): US 2001274270 P 20010307; US 98199120 A 19981124; US 200292047 A 20020304
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20020099729	A1		32	G06F-012/00	CIP of application US 98199120
					Provisional application US 2001274270
					CIP of patent US 6353836

Inventor: **BAMFORD R J** ...

... **BRIDGE W H** ...

... **BROWER D** ...

... **CHAN W W S** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V**

16/3,K/2 (Item 2 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014845124 **Image available**
WPI Acc No: 2002-665830/200271
Related WPI Acc No: 1999-494629; 2002-609207; 2002-636953; 2002-666065
XRPX Acc No: N02-526783

Cache managing method for multiple cache system, involves providing master coordinating with multiple caches to write data item in persistent storage in response to data writing request

Patent Assignee: BAMFORD R J (BAMF-I); BRIDGE W H (BRID-I); BROWER D (BROW-I); CHAN W W S (CHAN-I); CHANDRASEKARAN S (CHAN-I); MACNAUGHTON N (MACN-I); SRIHARI V (SRIH-I)

Inventor: **BAMFORD R J** ; **BRIDGE W H** ; **BROWER D** ; **CHAN W W S** ; **CHANDRASEKARAN S** ; **MACNAUGHTON N** ; **SRIHARI V**
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20020095403	A1	20020718	US 98199120	A	19981124	200271	B
			US 2001274270	A	20010307		
			US 200291618	A	20020304		

Priority Applications (No Type Date): US 2001274270 P 20010307; US 98199120 A 19981124; US 200291618 A 20020304
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20020095403	A1		34	G06F-007/00	CIP of application US 98199120
					Provisional application US 2001274270
					CIP of patent US 6353836

Inventor: **BAMFORD R J** ...

... **BRIDGE W H** ...

... **BROWER D** ...

... **CHAN W W S** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V**

16/3,K/3 (Item 3 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014816247 **Image available**
WPI Acc No: 2002-636953/200268
Related WPI Acc No: 1999-494629; 2002-339699; 2002-609207; 2002-666065
XRPX Acc No: N02-503192
Cache management techniques for use in systems with multiple caches holding different copies of the same data items, uses a master to coordinate with the multiple caches
Patent Assignee: ORACLE INT CORP (ORAC-N); INT BUSINESS MACHINES CORP (IBMC)
Inventor: **BAMFORD R** ; **BRIDGE W** ; **BROWER D** ; **CHAN W** ; **CHANDRASEKARAN S** ; **MACNAUGHTON N** ; **SRIHARI V** ; AMINI L; FROSSARD P; VANDERGHEYNST P; VERSCHEURE O
Number of Countries: 101 Number of Patents: 005
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 200273416	A2	20020919	WO 2002US6981	A	20020307	200268	B
US 6628300	B2	20030930	US 2001334514	P	20011130	200367	
			US 2001334521	P	20011130		
			US 200292247	A	20020306		
EP 1412858	A2	20040428	EP 2002717580	A	20020307	200429	
			WO 2002US6981	A	20020307		
AU 2002248570	A1	20020924	AU 2002248570	A	20020307	200433	

CN 1524226   A   20040825 CN 2002806161   A   20020307 200477

Priority Applications (No Type Date): US 200292247 A 20020304; US 2001274270 P 20010307; US 2001334514 P 20011130; US 2001334521 P 20011130
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 200273416	A2	E	57	G06F-012/00	

Designated States (National): AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ OM PH PL PT RO RU SD SE SG SI SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA ZM ZW
Designated States (Regional): AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZM ZW

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6628300	B2			G09G-005/00	Provisional application US 2001334514 Provisional application US 2001334521
EP 1412858	A2	E		G06F-012/00	Based on patent WO 200273416

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
AU 2002248570	A1			G06F-012/00	Based on patent WO 200273416
CN 1524226	A			G06F-012/00	

Inventor: **BAMFORD R** ...

... **BRIDGE W** ...

... **BROWER D** ...

... **CHAN W** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V**

16/3,K/4   (Item 4 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014788501   **Image available**
WPI Acc No: 2002-609207/200265
Related WPI Acc No: 1999-494629; 2002-339699; 2002-636953; 2002-666065
XRPX Acc No: N02-482415

Managing system caches by sending write notification message from master to node indicating persistent storage of most recent modified data item version

Patent Assignee: ORACLE INT CORP (ORAC-N); ORACLE CORP (ORAC-N); BAMFORD R J (BAMF-I); BRIDGE W H (BRID-I); BROWER D (BROW-I); CHAN W W S (CHAN-I); CHANDRASEKARAN S (CHAN-I); MACNAUGHTON N (MACN-I); SRIHARI V (SRIH-I)
Inventor: **BAMFORD R** ; **BRIDGE W** ; **BROWER D** ; **CHAN W** ; **CHANDRASEKARAN S** ; **MACNAUGHTON N** ; **SRIHARI V** ; **BAMFORD R J** ; **BRIDGE W H** ; **CHAN W W S**
Number of Countries: 101 Number of Patents: 006
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 200271229	A2	20020912	WO 2002US7475	A	20020306	200265	B
US 20020095403	A1	20020718	US 98199120	A	19981124	200271	
			US 2001274270	P	20010307		

```
                              US 200291618    A    20020304
EP 1366420     A2   20031203  EP 2002748408   A    20020306  200380
                              WO 2002US7475   A    20020306
AU 2002248570  A1   20020924  AU 2002248570   A    20020307  200433
AU 2002335503  A1   20020919  AU 2002335503   A    20020306  200433
CN 1496510     A    20040512  CN 2002806162   A    20020306  200452
                              WO 2002US7475   A    20020306

Priority Applications (No Type Date): US 200291618 A 20020304; US
  2001274270 P 20010307; US 98199120 A 19981124; US 200292247 A 20020304
Patent Details:
Patent No  Kind Lan Pg   Main IPC    Filing Notes
WO 200271229  A2 E  64 G06F-012/00
   Designated States (National): AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA
   CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN
   IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ
   OM PH PL PT RO RU SD SE SG SI SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA
   ZM ZW
   Designated States (Regional): AT BE CH CY DE DK EA ES FI FR GB GH GM GR
   IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZM ZW
US 20020095403 A1    34 G06F-007/00   CIP of application US 98199120
                                      Provisional application US 2001274270
                                      CIP of patent US 6353836
EP 1366420    A2 E     G06F-012/08    Based on patent WO 200271229
   Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT
   LI LT LU LV MC MK NL PT RO SE SI TR
AU 2002248570 A1       G06F-012/00    Based on patent WO 200273416
AU 2002335503 A1       G06F-012/00    Based on patent WO 200271229
CN 1496510    A        G06F-012/08
```

Inventor: **BAMFORD R** ...

... **BRIDGE W** ...

... **BROWER D** ...

... **CHAN W** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V** ...

... **BAMFORD R J** ...

... **BRIDGE W H** ...
... **CHAN W W S**
?

Scitech Bibliographic files

```
ds; show files

Set     Items   Description
S1    1390880   (CACHE? OR   TEMPORARY()(STORAGE OR   MEMORY? )
             OR BUFFER? OR RAM?)
S2    5109087   (WRIT??? OR WRIT???()DISK? OR TRANSFER??? OR RECORD??? OR -
             ENCOD??? OR UPDAT???)
S3     230224   (OLD OR ORIGINAL? OR PREVIOUS OR EARL??? OR OUTDAT???)(7N)-
             (NEW??? OR RECENT OR MODERN OR FRESH)
S4     118715   (MODIF???  OR MODIFICAT??? OR AMEND??? OR AMENDM??? OR CHA-
             NG??? OR ADJUST??? OR ADJUSTM???)(3N)(COP??? OR DATA OR FILE??
              OR INFORMATION?? OR INFO? OR VERSION? )
S5    6070322   (NODE? OR TERMINAL? OR COMPUTER? OR CLIENT? OR SERVER? OR -
             WORKSTATION?? OR STATION??)
S6      15297   (PERSIST??? OR PERSEVER??? OR ENDUR??? OR LINGER??? OR REM-
             AIN??? OR STEAD??? OR PROLONG??? OR UNCHANG???)(3N)(STOR??? OR
              STORAGE? OR MEMOR??? OR HARD()DRIVE OR DISK? OR CD()ROM?? OR
             CDROM OR ROM?? OR FLASH? OR EPROM?? OR PROM??)
S7     623623   (MASTER? OR CONTROL??? OR MAIN OR PRIMARY? OR SUPERVIS??? -
              OR ADMINISTRA??? OR MANAG?)(5N) S5
S8        839   AU=(CHANDRASEKARAN, S?  OR CHANDRASEKARAN S?)
S9        151   AU=(BAMFORD, R? OR BAMFORD R? )
S10        38   AU=(BRIDGE, W? OR  BRIDGE W?)
S11       673   AU=(BROWER, D? OR  BROWER D?)
S12        28   AU=(MACNAUGHTON, N?  OR MACNAUGHTON N?)
S13      7724   AU=(CHAN, W? OR  CHAN W?)
S14        14   AU=(SRIHARI, V? OR SRIHARI V?)
S15      9454   S8 OR S9 OR S10 OR S11 OR S12 OR S13 OR S14
S16         0   S8 AND S9 AND S10 AND S11 AND S12 AND S13 AND S14
S17         1   S15 AND S1 AND S2 AND S3
S18         0   S17 NOT DROSOPHILA
S19         0   S1 AND S2 AND S3 AND S4 AND S6 AND S7
S20      1037   S1 AND S2 AND S3
S21        20   S1 AND S2 AND S3 AND S4
S22       265   S6 AND S7
S23        22   S6(3N)S7
S24         6   S1 AND S2 AND S3 AND S6
S25        98   S1 AND S2 AND S3 AND S7
S26       329   S1 AND S2 AND S3 AND S5
S27         4   S1(10N)S2(10N)S3(10N)S7
S28        10   S1(25N)S2(25N)S3(25N)S7
S29        36   S1(10N)S2(10N)S3(10N)S5
S30        15   S1(7N)S2(7N)S3(7N)S5
S31        14   RD S21 (unique items)
S32        15   RD S23 (unique items)
S33         4   RD S24 (unique items)
S34         6   RD S28 (unique items)
S35        11   RD S30 (unique items)
S36        49   S31 OR S32 OR S33 OR S34 OR S35
S37        49   RD (unique items)
S38        32   S37 NOT PY>1999
S39        31   S38 NOT PD=19981124:20001124
S40        31   S39 NOT PD=20001124:20021124
S41        31   S40 NOT PD=20021124:20050103
File   2:INSPEC 1969-2004/Dec W2
         (c) 2004 Institution of Electrical Engineers
File   6:NTIS 1964-2004/Dec W4
         (c) 2004 NTIS, Intl Cpyrght All Rights Res
File   8:Ei Compendex(R) 1970-2004/Dec W3
         (c) 2004 Elsevier Eng.  Info. Inc.
```

```
File  34:SciSearch(R) Cited Ref Sci 1990-2004/Dec W4
        (c) 2004 Inst for Sci Info
File  65:Inside Conferences 1993-2004/Dec W4
        (c) 2004 BLDSC all rts. reserv.
File  94:JICST-EPlus 1985-2004/Nov W4
        (c)2004 Japan Science and Tech Corp(JST)
File  95:TEME-Technology & Management 1989-2004/Jun W1
        (c) 2004 FIZ TECHNIK
File  99:Wilson Appl. Sci & Tech Abs 1983-2004/Nov
        (c) 2004 The HW Wilson Co.
File 144:Pascal 1973-2004/Dec W1
        (c) 2004 INIST/CNRS
File 239:Mathsci 1940-2004/Feb
        (c) 2004 American Mathematical Society
File 434:SciSearch(R) Cited Ref Sci 1974-1989/Dec
        (c) 1998 Inst for Sci Info
File 583:Gale Group Globalbase(TM) 1986-2002/Dec 13
        (c) 2002 The Gale Group
File 603:Newspaper Abstracts 1984-1988
        (c)2001 ProQuest Info&Learning
File 483:Newspaper Abs Daily 1986-2004/Dec 31
        (c) 2005 ProQuest Info&Learning
File 248:PIRA 1975-2004/Dec W2
        (c) 2004 Pira International
File 438:Library Lit. & Info. Science 1984-2004/Oct
        (c) 2004 The HW Wilson Co
?
```

41/3,K/1 (Item 1 from file: 2)
DIALOG(R)File 2:INSPEC

6576766 INSPEC Abstract Number: C2000-06-6160S-006
Title: A high-performance spatial storage system based on main-memory database architecture
Author(s): Jang Ho Park; Kihong Kim; Cha, S.K.; Min Seok Song; Sangho Lee ; Juchang Lee
Author Affiliation: Knowledge & Data Eng. Lab., Seoul Nat. Univ., South Korea
Conference Title: Database and Expert Systems Applications. 10th International Conference, DEXA'99 (Lecture Notes in Computer Science Vol.1677) p.1066-75
Editor(s): Bench-Capon, T.; Soda, G.; Tjoa, A.M.
Publisher: Springer-Verlag, Berlin, Germany
Publication Date: 1999 Country of Publication: Germany xviii+1105 pp.
ISBN: 3 540 66448 3 Material Identity Number: XX-1999-02591
Conference Title: Proceedings of DEXA'99: 10th International Conference and Workshop on Database and Expert Systems Applications
Conference Date: 30 Aug.-3 Sept. 1999 Conference Location: Florence, Italy
Language: English
Subfile: C
Copyright 2000, IEE
...Abstract: to satisfy because most of them employ the traditional disk-based database architecture. With the **steadily** increasing **memory** capacity of **computer** systems, the **main** -memory database architecture becomes a feasible approach to meeting the requirement, and a few commercial...

41/3,K/2 (Item 2 from file: 2)
DIALOG(R)File 2:INSPEC

6428088 INSPEC Abstract Number: C2000-01-6160J-008
Title: Advanced conceptual clustering and associated querying facilities in object-oriented databases
Author(s): Chi-Wai Fung; Qing Li
Author Affiliation: Dept. of Comput. Studies, Hong Kong Inst. of Vocational Educ., Hong Kong
Journal: International Journal of Software Engineering and Knowledge Engineering vol.9, no.3 p.343-67
Publisher: World Scientific,
Publication Date: June 1999 Country of Publication: Singapore
CODEN: ISEKEW ISSN: 0218-1940
SICI: 0218-1940(199906)9:3L.343:ACCA;1-T
Material Identity Number: 0879-1999-004
Language: English
Subfile: C
Copyright 1999, IEE

...Abstract: and roles). An experimental prototype of the CCM has been constructed on top of a **persistent** object **storage** **manager** , running on a Sun4 **workstation** .

41/3,K/3 (Item 3 from file: 2)

DIALOG(R)File 2:INSPEC

5937750 INSPEC Abstract Number: B9807-6140C-267, C9807-5260B-167
Title: Picture quality editing for variable rate compressed video
Author(s): Yogeshwar, J.; Tsinberg, M.
Author Affiliation: Front Porch Video Inc., Lawrenceville, NJ, USA
Journal: SMPTE Journal vol.107, no.4 p.236-42
Publisher: Soc. Motion Picture & Telev. Eng,
Publication Date: April 1998 Country of Publication: USA
CODEN: SMPJDF ISSN: 0036-1682
SICI: 0036-1682(199804)107:4L.236:PQEV;1-C
Material Identity Number: S218-98005
Language: English
Subfile: B C
Copyright 1998, IEE

Abstract: This paper discusses a novel video **encoding** system in which a user controls the quality of the **encoded** video. It uses an algorithm for reallocation of target bits to the segment selected for...

... Bitstream splicing is carried out to complete the process of editing. The video is initially **encoded** according to an automatic process without user intervention. An operator then reviews it manually and...
... of the user-selected priority, The bit assignments are adjusted resulting in the avoidance of **buffer** underflow. Also, the **buffer**

occupancy at the trailing edge of the edit segment is maintained at its **original** level. After the assignment of **new** target bits, the average quantizer scale value is determined using a previously determined rate quantizer function. The video is then re- **encoded** with the new set of target bits to achieve the desired quality **change** . Bitstream and log **files** are **updated** by proper splicing.
...Identifiers: video **encoding** ;

41/3,K/4 (Item 4 from file: 2)
DIALOG(R)File 2:INSPEC

5767726 INSPEC Abstract Number: B9801-6150M-028, C9801-5640-022
Title: Optimal feedback control for ABR service in ATM
Author(s): Narvaez, P.; Kai-Yeung Siu
Author Affiliation: d'Arbeloff Lab. for Inf. Syst. & Technol., MIT, Cambridge, MA, USA
Conference Title: Proceedings. 1997 International Conference on Network Protocols (Cat. No.97TB100174) p.32-41
Editor(s): Ammar, M.; Shankar, U.
Publisher: IEEE Comput. Soc, Los Alamitos, CA, USA
Publication Date: 1997 Country of Publication: USA xi+309 pp.
ISBN: 0 8186 8061 X Material Identity Number: XX97-02841
U.S. Copyright Clearance Center Code: 0 8186 8061 X/97/$10.00
Conference Title: Proceedings 1997 International Conference on Network Protocols
Conference Sponsor: IEEE Comput. Soc. Tech. Committee on Distributed Process.; IEEE Commun. Soc. Tech. Committee on Comput. Commun
Conference Date: 28-31 Oct. 1997 Conference Location: Atlanta, GA, USA
Language: English

Subfile: B C

...Abstract: data traffic over ATM networks requires congestion control, whose objectives include maximizing throughput, minimizing switch **buffer** requirement, and attaining a fair bandwidth allocation. With available bit rate (ABR) service in ATM, congestion control is achieved by requiring **data** sources to **adjust** their rates based on the feedback from the network. The difficulties in providing effective ABR...

... the dynamic nature of the available bandwidth, as well as the feedback delay. Using a **new** design methodology described in our **previous** paper, we present here a congestion control algorithm that is provably stable and is optimal...
Descriptors: asynchronous **transfer** mode...
...Identifiers: switch **buffer** requirement...

41/3,K/5 (Item 5 from file: 2)
DIALOG(R)File 2:INSPEC

5528729 INSPEC Abstract Number: C9705-6160S-018
Title: VIMS: a video information management system
Author(s): John Chung-Mong Lee; Qing Li; Wei Xiong
Author Affiliation: Dept. of Comput. Sci., Hong Kong Univ. of Sci. & Technol., Kowloon, Hong Kong
Journal: Multimedia Tools and Applications vol.4, no.1 p.7-28
Publisher: Kluwer Academic Publishers,
Publication Date: Jan. 1997 Country of Publication: Netherlands
CODEN: MTAPFB ISSN: 1380-7501
SICI: 1380-7501(199701)4:1L.7:VVIM;1-X
Material Identity Number: D433-97001
U.S. Copyright Clearance Center Code: 1380-7501/97/$8.50
Language: English
Subfile: C

...Abstract: on semantic features/index terms. A prototype of this system has been constructed, using a **persistent** object **storage** **manager** (viz., EOS), on Sun4 **workstations** .

41/3,K/6 (Item 6 from file: 2)
DIALOG(R)File 2:INSPEC

5270044 INSPEC Abstract Number: C9607-6160S-012
Title: An experimental video database management system based on advanced object-oriented techniques
Author(s): Liusheng Huang; Lee, J.C.-M.; Qing Li; Wei Xiong
Author Affiliation: Dept. of Comput. Sci., Univ. of Sci. & Technol., Kowloon, Hong Kong
Journal: Proceedings of the SPIE - The International Society for Optical Engineering Conference Title: Proc. SPIE - Int. Soc. Opt. Eng. (USA) vol.2670 p.158-69
Publisher: SPIE-Int. Soc. Opt. Eng,
Publication Date: 1996 Country of Publication: USA
CODEN: PSISDG ISSN: 0277-786X

SICI: 0277-786X(1996)2670L.158:EVDM;1-P
Material Identity Number: C574-96080
U.S. Copyright Clearance Center Code: 0 8194 2044 1/96/$6.00
Conference Title: Storage and Retrieval for Still Image and Video Databases IV
Conference Sponsor: SPIE; Soc. Imaging Sci. & Technol
Conference Date: 1-2 Feb. 1996 Conference Location: San Jose, CA, USA
Language: English
Subfile: C

...Abstract: on semantic features/index terms. A prototype of this system has been constructed, using a **persistent** object **storage manager** (viz. EOS), on Sun4 **workstations** .

41/3,K/7 (Item 7 from file: 2)
DIALOG(R)File 2:INSPEC

4923357 INSPEC Abstract Number: C9505-6110P-030
Title: PPOST: a parallel database in main memory
Author(s): Boszormenyi, L.; Eder, J.; Weich, C.
Author Affiliation: Inst. fur Inf., Klagenfurt Univ., Austria
p.754-8
Editor(s): Karagiannis, D.
Publisher: Springer-Verlag, Berlin, Germany
Publication Date: 1994 Country of Publication: West Germany xvii+807 pp.
ISBN: 3 540 58435 8
Conference Title: Proceedings of DEXA 94
Conference Date: 7-9 Sept. 1994 Conference Location: Athens, Greece
Language: English
Subfile: C

Abstract: We present the PPOST architecture (**Persistent** Parallel Object **Store**) for **main** memory database systems on parallel **computers** , that is suited for applications with challenging performance requirements. The architecture takes full advantage of...

41/3,K/8 (Item 8 from file: 2)
DIALOG(R)File 2:INSPEC

04396633 INSPEC Abstract Number: C9306-5220-009
Title: Floating parity and data disk **arrays**
Author(s): Menon, J.; Roche, J.; Kasson, J.
Author Affiliation: IBM Almaden Res. Center, San Jose, CA, USA
Journal: Journal of Parallel and Distributed Computing vol.17, no.1-2 p.129-39
Publication Date: Jan.-Feb. 1993 Country of Publication: USA
CODEN: JPDCER ISSN: 0743-7315
U.S. Copyright Clearance Center Code: 0743-7315/93/$5.00
Language: English
Subfile: C

Title: Floating parity and data disk **arrays**

Abstract: A **disk** array is a set of **disk** drives (and controller) which can automatically recover data when one or more **disk** drives in the set fail. One method used by **disk** arrays to achieve high availability at lower cost than mirroring is a parity technique. This paper considers **disk** arrays that use the parity technique. The main drawback of such arrays is that they need four **disk** accesses to **update** a data block-two to read **old** data and parity, and two to **write** **new** data and parity. The authors describe four new methods to improve the **update** performance of **disk** arrays that use the parity technique from four accesses to three and, in some cases, to two. All their schemes sacrifice **disk** storage efficiency for improved **update** performance by relaxing the requirement that the **modified** **data** and parity blocks be **written** back into their original locations. Their best technique, called floating parity track, achieves much improved **update** performance while using only 1% more **disk** space than traditional arrays.

Identifiers: data **disk** arrays...

... **disk** drives...

... **update** performance...

... **disk** storage efficiency

41/3,K/9 (Item 9 from file: 2)
DIALOG(R)File 2:INSPEC

04244216 INSPEC Abstract Number: C9211-5320C-007
Title: Methods for improved update **performance of** disk **arrays**
Author(s): Menon, J.; Kasson, J.
Author Affiliation: IBM Almaden Res. Center, San Jose, CA, USA
Conference Title: Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences (Cat. No.91TH0394-7) p.74-83 vol.1
Editor(s): Milutinovic, V.; Shriver, B.D.; Nunamaker, J.F., Jr.; Sprague, R.H., Jr.
Publisher: IEEE Comput. Soc. Press, Los Alamitos, CA, USA
Publication Date: 1991 Country of Publication: USA 4 vol. (xv+831+xv+877+xii+670+xiii+729) pp.
ISBN: 0 8186 2420 5
U.S. Copyright Clearance Center Code: 0073-1129/92$3.00
Conference Sponsor: Univ. Hawaii; ACM; IEEE; Pacific Res. Inst. Inf. Syst. Manage
Conference Date: 7-10 Jan. 1992 Conference Location: Kauai, HI, USA
Language: English
Subfile: C

Title: Methods for improved update **performance of** disk **arrays**
Abstract: A **disk** array is a set of **disk** drives (and controller) which can automatically recover data when one (or more) **disk** drives in the set fails. One method used by **disk** arrays to achieve high availability at lower cost than mirroring is a parity technique. The main drawback of such arrays are that they need four **disk** accesses to **update** a data block-two to read **old** data and parity, and two to **write** **new** data and parity. The authors describe four new methods to improve the **update** performance of **disk** arrays that use the parity technique from four accesses to three and, in some cases, to two. All the schemes sacrifice **disk** storage

efficiency for improved **update** performance by relaxing the requirement that the **modified data** and parity blocks be **written** back into their original locations. The best technique, called 'floating parity track', achieves much improved **update** performance while using only 1% more **disk** space than traditional arrays.
...Identifiers: **update** performance...

... **disk** arrays...

... **disk** drives...

... **disk** accesses...

... **disk** storage efficiency...
... **modified data** ;

41/3,K/10 (Item 10 from file: 2)
DIALOG(R)File 2:INSPEC

03181778 INSPEC Abstract Number: C88048361
Title: Developing software for freshman composition: Sentence Patterns
Author(s): Cox, D.
Author Affiliation: Dept. of English, Amarillo Coll., TX, USA
Journal: Collegiate Microcomputer vol.6, no.2 p.161-4
Publication Date: May 1988 Country of Publication: USA
CODEN: CMICDL ISSN: 0731-4213
Language: English
Subfile: C

...Abstract: basic reasons: to instruct students to use sentences with clarity, conciseness, and variety in their **writing** ; and to provide a **new** method- **computer** software-to solve an **old** problem, the effective use of **written** language. After development and testing of the software, student surveys showed that the **disks** were easy to use, time saving, and helpful. This article deals with the rationale for...

41/3,K/11 (Item 11 from file: 2)
DIALOG(R)File 2:INSPEC

02625306 INSPEC Abstract Number: C86019445
Title: Teamwork-method for quick reference on actual data
Author(s): Schroeder, K.
Conference Title: CAMP '85. Computer Graphics: Applications for Management and Productivity. European Conference and Exhibition p.116
Publisher: AMK Berlin, Berlin, West Germany
Publication Date: 1985 Country of Publication: West Germany 453 pp.
Conference Sponsor: World Comput. Graphics Assoc
Conference Date: 24-27 Sept. 1985 Conference Location: Berlin, West Germany
Language: English
Subfile: C

...Abstract: version of the information, handed over to the mass storage of a computer. Appending and **changing** of this **data** runs under control

of the SCCS. Document administration kept by SCCS, is without any risk...

... getting access to the different versions created within the lifetime of a special document. The **disk** space occupied by data is very low, especially compared to the **old** fashioned method, creating the **new** version by copying the **old** one.
...Descriptors: **records** management
...Identifiers: **disk** space

41/3,K/12 (Item 12 from file: 2)
DIALOG(R)File 2:INSPEC

02302450 INSPEC Abstract Number: C84039981, D84002380
Title: Trendtext/2 on the IBM PC
Author(s): Townsend, K.
Journal: Mind Your Own Business vol.7, no.7 p.49-50
Publication Date: July-Aug. 1984 Country of Publication: UK
CODEN: MYOBD4 ISSN: 0143-1374
Language: English
Subfile: C D

...Abstract: editor. This means that any text file can be as large as the available space **remaining** on the **disk** : as the **computer** 's **main** memory fills with text, the system automatically transfers the overflow on to the disk, thus...

41/3,K/13 (Item 13 from file: 2)
DIALOG(R)File 2:INSPEC

00466286 INSPEC Abstract Number: C73001763
Title: Quantitative aspects in the use of indexed sequential file organisation
Author(s): van Hoeve, F.A.; Koot, C.W.M.
Journal: Informatie vol.14, no.7-8 p.358-68
Publication Date: July-Aug. 1972 Country of Publication: Netherlands
CODEN: INFTCR ISSN: 0019-9907
Language: Dutch
Subfile: C

...Abstract: generating new master files are compared: a 'copy' method, viz., where the original 'master' is **copied** with **changes** incorporated on to a **new** master, and a 'direct' method where the **original** 'master' is overwritten with the **changes** , destroying the original **information** . Advantages and disadvantages of each type of operation are given, as are work times, access times, **buffer** size, etc.
...Identifiers: **buffer** size...

... **updating**

41/3,K/14 (Item 1 from file: 6)
DIALOG(R)File 6:NTIS

0928059 NTIS Accession Number: PB82-801291/XAB

Highway Ramp Control. 1964-October, 1981 (Citations from the NTIS Data Base)

(Rept. for 1964-Oct 81)
National Technical Information Service, Springfield, VA.
Corp. Source Codes: 055665000
Nov 81 162p
Languages: English Document Type: Bibliography
Journal Announcement: GRAI8203
Supersedes PB80-812803, and NTIS/PS-79/0644.
Order this product from NTIS by: phone at 1-800-553-NTIS (U.S. customers); (703)605-6000 (other countries); fax at (703)321-8547; and email at orders@ntis.fedworld.gov. NTIS is located at 5285 Port Royal Road, Springfield, VA, 22161, USA.
NTIS Prices: PC N01/MF N01

The bibliography contains citations of reports on ramp design, merging control, ramp metering, and **ramp** use. Studies are documented of freeway collector lanes, exits into city streets, traffic bottlenecks, express bus lanes, vehicle collisions, road curves, freeway entry **control** --all involving **ramp** engineering. **Computerized** simulations and freeway corridor models are presented, along with traffic models, **ramp** control strategies, optimization procedures, and studies of freeway traffic flow. A few cases in which accident incidence has been linked to faulty **ramp** design, thereby calling for improved vehicle monitoring has been included. (This **updated** bibliography contains 156 citations, 14 of which are **new** entries to the **previous** edition.)

41/3,K/15 (Item 2 from file: 6)
DIALOG(R)File 6:NTIS

0835344 NTIS Accession Number: PB80-812803/XAB

Highway Ramp Control. 1964-June, 1980 (A Bibliography with Abstracts)

(Rept. for 1964-Jun 80)
Kenton, E.
National Technical Information Service, Springfield, VA.
Corp. Source Codes: 055665000
Jul 80 149p
Languages: English Document Type: Bibliography
Journal Announcement: GRAI8020
Supersedes NTIS/PS-79/0644 and NTIS/PS-78/0588.
Order this product from NTIS by: phone at 1-800-553-NTIS (U.S. customers); (703)605-6000 (other countries); fax at (703)321-8547; and email at orders@ntis.fedworld.gov. NTIS is located at 5285 Port Royal Road, Springfield, VA, 22161, USA.
NTIS Prices: PC N01/MF N01

The bibliography contains citations of reports on ramp design, merging control, ramp metering, and **ramp** use. Studies are documented of freeway collector lanes, exits into city streets, traffic bottlenecks, express bus lanes, vehicle collisions, road curves, freeway entry **control** --all involving **ramp** engineering. **Computerized** simulations and freeway corridor models are presented, along with traffic models, **ramp** control strategies, optimization procedures, and studies of freeway traffic flow. A few cases in which accident incidence has been linked to faulty **ramp** design, thereby calling for improved vehicle monitoring has been included.

(This **updated** bibliography contains 142 abstracts, 12 of which are **new** entries to the **previous** edition.)

41/3,K/16 (Item 3 from file: 6)
DIALOG(R)File 6:NTIS

0765923 NTIS Accession Number: NTIS/PS-79/0644/9/XAB
Highway Ramp Control (A Bibliography with Abstracts)
(Rept. for 1964-Jun 79)
Kenton, E.
National Technical Information Service, Springfield, VA.
Corp. Source Codes: 391812
Jul 79 130p
Languages: English Document Type: Bibliography
Journal Announcement: GRAI7918
Supersedes NTIS/PS-78/0588 and NTIS/PS-77/0491.
Order this product from NTIS by: phone at 1-800-553-NTIS (U.S. customers); (703)605-6000 (other countries); fax at (703)321-8547; and email at orders@ntis.fedworld.gov. NTIS is located at 5285 Port Royal Road, Springfield, VA, 22161, USA.
NTIS Prices: PC N01/MF N01

The bibliography contains citations of reports on ramp design, merging control, ramp metering, and **ramp** use. Studies are documented of freeway collector lanes, exits into city streets, traffic bottlenecks, express bus lanes, vehicle collisions, road curves, freeway entry **control** --all involving **ramp** engineering. **Computerized** simulations and freeway corridor models are presented, along with traffic models, **ramp** control strategies, optimization procedures, and studies of freeway traffic flow. A few cases in which accident incidence has been linked to faulty **ramp** design, thereby calling for improved vehicle monitoring has been included. (This **updated** bibliography contains 130 abstracts, 26 of which are **new** entries to the **previous** edition.)

41/3,K/17 (Item 4 from file: 6)
DIALOG(R)File 6:NTIS

0692737 NTIS Accession Number: AD-A052 432/2/XAB
IC Fabrication Using Electron-Beam Technology
(Quarterly rept. no. 1, 1 Jul-1 Oct 76)
Varnell, G. L. ; Williamson, R. A. ; Brewer, T. L. ; Bartelt, J. L. ; Dexter, R. J.
Texas Instruments Inc Dallas
Corp. Source Codes: 347650
Report No.: TI-03-77-02
Jan 77 46p
Journal Announcement: GRAI7814
Order this product from NTIS by: phone at 1-800-553-NTIS (U.S. customers); (703)605-6000 (other countries); fax at (703)321-8547; and email at orders@ntis.fedworld.gov. NTIS is located at 5285 Port Royal Road, Springfield, VA, 22161, USA.
NTIS Prices: PC A03/MF A01

A program to implement e-beam **writing** technology for fabrication of

microcircuits has been initiated this quarter. The technical and economic impact of electron-beam direct slice printing will be demonstrated on 256-bit bipolar **RAMs** utilizing an electron-beam machine (EBMIIIA). **Earlier** this year a **new** high-speed interface was installed on EBMIII to allow up to 5 micrometers/microsec scan...

...requires only 8 address lines to select one of 256 storage locations. An additional line, **write** enable, is provided to enable the memory to **modify** the stored **data** . Separate Data Input and Data Output lines are provided for minimum interaction between input and...

41/3,K/18 (Item 1 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

04650252 E.I. No: EIP97033574851
Title: VIMS: a video information management system
Author: Lee, John Chung-Mong; Li, Qing; Xiong, Wei
Corporate Source: Hong Kong Univ of Science and Technology, Kowloon, Hong Kong
Source: Multimedia Tools and Applications v 4 n 1 Jan 1997. p 7-28
Publication Year: 1997
CODEN: MTAPFB ISSN: 1380-7501
Language: English

...Abstract: on semantic features/index terms. A prototype of this system has been constructed, using a **persistent** object **storage** **manager** (viz., EOS), on Sun4 **workstations** . (Author abstract) 29 Refs.

41/3,K/19 (Item 2 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

04399084 E.I. No: EIP96043116189
Title: Experimental video database management system based on advanced object-oriented techniques
Author: Huang, Liusheng; Lee, John C.; Li, Qing; Xiong, Wei
Corporate Source: Hong Kong Univ. of Science and Technology, Hong Kong, Hong Kong
Conference Title: Storage and Retrieval for Still Image and Video Databases IV
Conference Location: San Jose, CA, USA Conference Date: 19960201-19960202
E.I. Conference No.: 22496
Source: Proceedings of SPIE - The International Society for Optical Engineering v 2670 1996. Society of Photo-Optical Instrumentation Engineers, Bellingham, WA, USA. p 158-169
Publication Year: 1996
CODEN: PSISDG ISSN: 0277-786X ISBN: 0-8194-2044-1
Language: English

...Abstract: on semantic features/index terms. A prototype of this system has been constructed, using a **persistent** object **storage** **manager** (viz. EOS), on Sun4 **workstations** . 16 Refs.

41/3,K/20 (Item 3 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

03864070 E.I. No: EIP94051293856
Title: Storage reclamation and reorganization in client-server persistent object stores
Author: Yong, Voon-Fee; Naughton, Jeffrey F.; Yu, Jie-Bing
Corporate Source: Univ of Wisconsin, Madison, WI, USA
Conference Title: Proceedings of the 10th International Conference on Data Engineering
Conference Location: Houston, TX, USA Conference Date: 19940214-19940218
E.I. Conference No.: 20218
Source: Proceedings - International Conference on Data Engineering 1994. Publ by IEEE, Computer Society Press, Los Alamitos, CA, USA,93CH3383-7. p 120-131
Publication Year: 1994
CODEN: PIDEEG ISBN: 0-8186-5400-7
Language: English

Identifiers: **Storage** reclamation; **Storage** reorganization; Client **server** **persistent** object **stores** ; Exodus **storage** **manager** ; Incremental partitioned collector; Transaction semantics; Reclustering data

41/3,K/21 (Item 4 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

03323229 E.I. Monthly No: EI9111133837
Title: Entering a new phase.
Author: Ryan, Bob
Corporate Source: BYTE, Peterborough, NH, USA
Source: Byte v 15 n 12 Nov 1990 p 289
Publication Year: 1990
CODEN: BYTEDJ ISSN: 0360-5280
Language: English

...Abstract: Phase-change technology has existed since the 1960s and is used in many commercial WORM (**write** once, read many times) drives. Now it has become rewritable. Phase-change technology is the first read/ **write** optical storage technology that allows for the direct overwriting of **old** data by **new** . By offering the high capacity and removability of optical media while eliminating the need to erase the media before **writing** , phase change holds a definite theoretical advantage over MO **storage** . It **remains** to be seen whether Matsushita and others can translate this advantage into superior products.
...Descriptors: **Disk** ; DATA STORAGE, MAGNETIC...

... **Disk** ; COMPUTER PERIPHERAL EQUIPMENT...
... **Disk** Drives; COMPUTER HARDWARE
Identifiers: PHASE CHANGE TECHNOLOGY; MAGNETO OPTICAL (MO) STORAGE; **WRITE** ONCE READ MANY TIMES (WORM)

41/3,K/22 (Item 5 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

00302577 E.I. Monthly No: EI7305023883 E.I. Yearly No: EI73023149
Title: HYBRID SIMULATION OF A SINGLE-SHAFT GAS TURBINE WATER PUMP DRIVE.
Author: Schatborn, I. W.
Corporate Source: Werkspoor-Amsterdam, Neth
Source: American Society of Mechanical Engineers (Paper) n 73-GT-72 for Meet Apr 8-12 1973 16 p
Publication Year: 1973
CODEN: ASMSA4 ISSN: 0402-1215
Language: ENGLISH

...Abstract: load within plant operating limitations. For this simulation work, the hybrid computer favorably combines analog **computer** flexibility in the solving **control** problems with digital **computer** **memory** for **storage** of **steady** -state characteristics. The latter are generated using a new hybrid function generation technique for two...

41/3,K/23 (Item 1 from file: 34)
DIALOG(R)File 34:SciSearch(R) Cited Ref Sci

06736285 Genuine Article#: ZN785 No. References: 10
Title: Research-oriented image registry for multimodal image integration
Author(s): Tanaka M (REPRINT) ; Sadato N; Ishimori Y; Yonekura Y; Yamashita Y; Komuro H; Hayahsi N; Ishii Y
Corporate Source: FUKUI MED SCH,DEPT RADIOL, 23 SHIMOAIZUKI, MATSUOKA CHO/FUKUI 91011//JAPAN/ (REPRINT); FUKUI MED SCH,BIOMED IMAGING RES CTR/FUKUI 91011//JAPAN/; FUKUI MED SCH,CTR MED INFORMAT/FUKUI 91011//JAPAN/
Journal: MEDICAL INFORMATICS, 1998, V23, N1 (JAN-MAR), P85-88
ISSN: 0307-7640 Publication date: 19980100
Publisher: TAYLOR & FRANCIS LTD, ONE GUNPOWDER SQUARE, LONDON EC4A 3DE, ENGLAND
Language: English Document Type: ARTICLE (ABSTRACT AVAILABLE)

...Abstract: patient. DDS informs the user through e-mail that new data have been generated and **transferred** . Data format is automatically converted into one which is chosen by the user. Data inactive for a certain period in the intermediate **server** are automatically archived into the final and permanent data **server** based on compact **disk** . As a soft link is automatically generated through this step, a user has access to all (**old** or **new**) image data of the patient of his interest. As DDS runs with minimal maintenance, cost and time for data **transfer** are significantly saved. By making the complex process of data transfer and conversion invisible, DDS...

41/3,K/24 (Item 2 from file: 34)
DIALOG(R)File 34:SciSearch(R) Cited Ref Sci

04503700 Genuine Article#: TH383 No. References: 55
Title: OLIGOMERIZATION IN AS(III) SULFIDE SOLUTIONS - THEORETICAL CONSTRAINTS AND SPECTROSCOPIC EVIDENCE
Author(s): HELZ GR; TOSSELL JA; CHARNOCK JM; PATTRICK RAD; VAUGHAN DJ; GARNER CD

Corporate Source: UNIV MARYLAND,WATER RESOURCES RES CTR/COLLEGE
PK//MD/20742; UNIV MANCHESTER/MANCHESTER M13 9PL/LANCS/ENGLAND/
Journal: GEOCHIMICA ET COSMOCHIMICA ACTA, 1995, V59, N22 (NOV), P4591-4604
ISSN: 0016-7037
Language: ENGLISH   Document Type: ARTICLE   (Abstract Available)

...Abstract: molecules and ions are predicted from molecular orbital theory and used to interpret EXAFS and **Raman** spectra of dissolved thioarsenites in undersaturated, alkaline 1 M NaHS solutions. From MO predictions, **Raman** peaks at 325 and 412 cm(-1) are assigned to AsS(SH)(2)(-) and a...

...thioarsenites are 2.21-2.23 Angstrom and no statistically significant As-As interactions are **recorded** , consistent with predominance of the monomers, AsS(SH)(2)(-) and AsS2(SH)(2-). Estimated proton...
...greater than 2 are unstable in water. In seeming contradiction to this spectroscopic evidence, a **new** analysis of published solubility studies reinforces **previous** inferences that the trimer, As3S4(SH)(2)(-), is the predominant thioarsenite in systems saturated with...
...Identifiers--EXISTING EXPERIMENTAL- **DATA** ; PHOTOSTRUCTURAL **CHANGES** ; BISULFIDE COMPLEXES; AQUEOUS-SOLUTION; NMR-SPECTRA; SOLUBILITY; SPECIATION; STABILITY; AS2S3; STOICHIOMETRY

41/3,K/25 (Item 3 from file: 34)
DIALOG(R)File 34:SciSearch(R) Cited Ref Sci

01019751 Genuine Article#: FP141 No. References: 10
Title: HERBRECS - THE QUEENSLAND HERBARIUM RECORDS SYSTEM - ITS DEVELOPMENT AND USE
Author(s): JOHNSON RW
Corporate Source: QUEENSLAND HERBARIUM,AGR RES LABS,MEIERS
RD/INDOOROOPILLY/QLD 4068/AUSTRALIA/
Journal: TAXON, 1991, V40, N2, P285-300
Language: ENGLISH   Document Type: ARTICLE   (Abstract Available)

...Abstract: will be to ensure that information associated with herbarium specimens, in particular the botanical name, **remains** identical to that **stored** on the **computer** **masterfile** while collections are being revised.

41/3,K/26 (Item 1 from file: 95)
DIALOG(R)File 95:TEME-Technology & Management

00726550 I93103997286
Formal specification of a persistent object management system
(Formale Spezifikation eines dauerhaften Objektverwaltungssystems)
Murphy, J; Grimson, J
Sch. of Comput. Appl., Fac. of Comput. & Math. Sci., Dublin City Univ., Ireland
Information and Software Technology, v35, n5, pp277-286, 1993
Document type: journal article Language: English
Record type: Abstract
ISSN: 0950-5849

IDENTIFIERS: SOFTWARE DESIGN; OBJECT ORIENTED DATABASE SYSTEM; PERSISTENT OBJECT MANAGEMENT SYSTEM; **PERSISTENT** OBJECT **STORAGE** **MANAGE** ; POSM; Z NOTATION; SCHEMAS; **NODE** **MANAGEMENT** COMPONENT; DATENBANKSCHEMA; objektorientierte Datenbank; formale Spezifikation

41/3,K/27 (Item 1 from file: 144)
DIALOG(R)File 144:Pascal

10866180 PASCAL No.: 93-0375544
Floating parity and data risk arrays
Parallel I/O systems
MENON J; ROCHE J; KASSON J
CHOUDHARY A, ed
IBM, Almaden res. cent., San Jose CA 95120-6099, USA
Syracuse univ., dep. electrical computer eng., Syracuse NY 13244, USA
Journal: Journal of parallel and distributed computing, 1993, 17 (1-2) 129-139
Language: English

A **disk** array is a set of **disk** drives (and controller) which can automatically recover data when one or more **disk** drives in the set fail. One method used by **disk** arrays to achieve high availability at lower cost than mirroring is a parity technique. This paper considers **disk** arrays that use the parity technique. The main drawback of such arrays is that they need four **disk** accesses to **update** a data block-two to read **old** data and parity, and two to **write** **new** data and parity. We describe four new methods to improve the **update** performance of **disk** arrays that use the parity technique from four accesses to three and, in some cases, to two. All our schemes sacrifice **disk** storage efficiency for improved **update** performance by relaxing the requirement that the **modified** **data** and parity blocks be **written** back into their original locations

English Descriptors: Storage system; Parallel **disk** ; Fault tolerance; **Updating** ; Performance; Input output

41/3,K/28 (Item 1 from file: 239)
DIALOG(R)File 239:Mathsci

02809899 MR 98h#00042
From universal morphisms to megabytes: a Baayen space odyssey.
On the occasion of the retirement of P. C. Baayen. Edited by Krzysztof Apt, Lex Schrijver and Nico Temme.
Contributors: Apt, Krzysztof; Schrijver, Lex; Temme, Nico; Baayen, P. C.
Publ: Stichting Mathematisch Centrum, Centrum voor Wiskunde en Informatica, Amsterdam,
1994, xii+642 pp. ISBN: 90-6196-450-4
Language: English
From universal morphisms to megabytes: a Baayen space odyssey; Biography: Baayen, P. C.; Bibliography: Baayen, P. C.; Festschrift: Baayen, P. C.
Subfile: MR (Mathematical Reviews) AMS
Abstract Length: LONG (66 lines)
Reviewer: Editors

...A. A. M. Kuijk, P. C. Marais and E. H. Blake, Adaptive spline-wavelet image **encoding** and real-time synthesis on a VLSI difference engine for image generation (405--419); Walter...

...and Jan H. van Schuppen, Control of discrete event systems---research at the interface of **control** theory and **computer** science (453--467).
Arjan Pellenkoft, Cesar Galindo-Legaria and Martin Kersten, Fast, randomized join-order...

...te Riele, Job scheduling on a parallel shared memory bus computer (485--491); Alexander Schrijver, **Rambling** along paths, trees, flows, curves, knots, and rails (493--534); Arno Siebes, Data mining: exploratory data analysis on very large databases (535--557); N. M. Temme, Bernoulli polynomials **old** and **new** : problems in complex analysis and asymptotics (559--576); A. S. Troelstra, Kleene's realizability (577...

41/3,K/29 (Item 1 from file: 583)
DIALOG(R)File 583:Gale Group Globalbase(TM)

04104472
PANASONIC LAUNCHES LF-7010 OPTICAL **DISK** DRIVE
JAPAN - PANASONIC LAUNCHES LF-7010 OPTICAL **DISK** DRIVE
Office Equipment & Products (OEP) 0 February 1991 p50
ISSN: 0387-5245

PANASONIC LAUNCHES LF-7010 OPTICAL **DISK** DRIVE
JAPAN - PANASONIC LAUNCHES LF-7010 OPTICAL **DISK** DRIVE

Panasonic Communications & Systems has launched the LF-87010, a multifunctional drive featuring phase- **change** technology allowing **data** to be over- **written** direct to a rewritable optical **disk** in one pass. The technology employs a single laser beam to **write** **new** information over **old** data in the same pass without the need for the erasure cycle of conventional methods...

... also features an embedded SCSI-II, an average seek time of 90 ms, maximum data **transfer** rate of 10.3 Mbps and a MTBF of 20k hours. It is fully compatible with Panasonics WORM **disks** .

41/3,K/30 (Item 1 from file: 248)
DIALOG(R)File 248:PIRA

00472359 Pira Acc. Num.: 20074672
Title: CD ReWritable should benefit publishers
Authors: Anon
Source: Digital Publ. Technol. vol. 1, no. 12, Dec. 1996, p. 3
ISSN: 1365-067X
Publication Year: 1996
Document Type: Journal Article
Language: English

...Abstract: In late October 1996 Hewlett Packard, Mitsubishi, Philips, Ricoh, Sony and Verbatim jointly unveiled the **new** format. The first equipment should ship in **early** 1997. CD RW discs can be **written** -over

repeatedly. Hewlett Packard expects them to replace 3.5in floppy **disks** and hard **disk** backup devices in everyday computing, and for this reason to become standard on PCs. It...

... for short run publishing and situations calling for an audit trail of unchangeable data, while **CD ROM** will **remain** for publishing in larger quantities. Those behind CD RW have specified specifications necessary to make...
... drives compatible with the new medium, but it remains to be seen how quickly that **transfers** into reality. (Short article)

41/3,K/31 (Item 2 from file: 248)
DIALOG(R)File 248:PIRA

00217482 Pira Acc. Num.: 9680927 Pira Abstract Numbers: 08-91-PT00273
Title: MICROCOLOR INK CONTROL AUTOMATION
Authors: Anon
Source: Graphix July 1990, pp 31, 33
Publication Year: 1990
Document Type: Journal Article
Language: English

Abstract: US Graphic Microsystems Inc. offer the Microcolor **computerised** ink **control** system, enabling fully automated ink duct control to be retrofitted to any press. With the...

...settings may be preset, calibrated adjustments made, and settings stored in memory or on floppy **disk** , all from the user-friendly console. The console provides push-button displays, and communicates with the press via simple prompts. Microcolor may operate independent of the press; a **new** job may be set up while the **old** one is running, or whilst the press undergoes cleaning. With an optional plate scanner, preset information from the scanner, located in the plate room, may be **transferred** to floppy **disk** , and then read by Microcolor. System construction is modular.
?

Patent Bibliographic files

```
? ds; show files

Set      Items   Description
S1      749655   (CACHE? OR   TEMPORARY()(STORAGE OR  MEMORY? )   OR
              OR BUFFER? OR RAM?)
S2     2226829   (WRIT??? OR WRIT???()DISK? OR TRANSFER??? OR RECORD??? OR -
              ENCOD??? OR UPDAT???)
S3       12118   (OLD OR ORIGINAL? OR PREVIOUS OR EARL??? OR OUTDAT???)(7N)-
              (NEW??? OR RECENT OR MODERN OR FRESH)
S4       75789   (MODIF???  OR MODIFICAT??? OR AMEND??? OR AMENDM??? OR CHA-
              NG??? OR ADJUST??? OR ADJUSTM???)(3N)(COP??? OR DATA OR FILE??
               OR INFORMATION?? OR INFO? OR VERSION? )
S5     1946417   (NODE? OR TERMINAL? OR COMPUTER? OR CLIENT? OR SERVER? OR -
              WORKSTATION?? OR STATION??)
S6       11093   (PERSIST??? OR PERSEVER??? OR ENDUR??? OR LINGER??? OR REM-
              AIN??? OR STEAD??? OR PROLONG??? OR UNCHANG???)(3N)(STOR??? OR
               STORAGE? OR MEMOR??? OR HARD()DRIVE OR DISK? OR CD()ROM?? OR
              CDROM OR ROM?? OR FLASH? OR EPROM?? OR PROM??)
S7      280446   (MASTER? OR CONTROL??? OR MAIN OR PRIMARY? OR SUPERVIS??? -
              OR ADMINISTRA??? OR MANAG?)(5N) S5
S8          37   AU=(CHANDRASEKARAN, S?  OR CHANDRASEKARAN S?)
S9          46   AU=(BAMFORD, R? OR BAMFORD R? )
S10         44   AU=(BRIDGE, W? OR  BRIDGE W?)
S11         38   AU=(BROWER, D? OR  BROWER D?)
S12          7   AU=(MACNAUGHTON, N?  OR MACNAUGHTON N?)
S13        294   AU=(CHAN, W? OR  CHAN W?)
S14          8   AU=(SRIHARI, V? OR SRIHARI V?)
S15          0   S1 AND S2 AND A3 AND S4 AND S5 AND S6 AND S7
S16        538   S1 AND S2 AND S3
S17         43   S1 AND S2 AND S3 AND S4
S18        303   S6 AND S7
S19          4   S1 AND S2 AND S3 AND S6
S20        161   S1 AND S2 AND S3 AND S5
S21         26   S1 AND S2 AND S3 AND S7
S22    1159387   IC=G06F?
S23          1   S19 AND S22
S24         65   S17 OR S21
S25         38   S24 AND S22
S26         31   S25 NOT PY>1999
S27         31   S26 NOT AD=19981124:19991124
S28         31   S27 NOT AD=19991124:20001124
S29         31   S28 NOT AD=20011124:20021124
S30         31   S29 NOT AD=20031124:20041230
S31         31   IDPAT (sorted in duplicate/non-duplicate order)
S32         31   IDPAT (primary/non-duplicate records only)
S33         71   (S1(10N)S2(10N) S3(10N)S5) OR (S6(10N)S7)
S34         53   S33 AND S22
S35         54   (S1(7N)S2(7N) S3(7N)S5) OR (S6(7N)S7)
S36         41   S35 AND S22
S37         36   S36 NOT (S32 OR S23)
S38          9   S37 NOT PY>1999
S39          9   S38 NOT AD=19981124:19991124
S40          9   S39 NOT AD=19991124:20001124
S41          9   S40 NOT AD=20011124:20021124
S42          9   S41 NOT AD=20031124:20041230
S43          9   IDPAT (sorted in duplicate/non-duplicate order)
S44          9   IDPAT (primary/non-duplicate records only)
File 347:JAPIO Nov 1976-2004/Aug(Updated 041203)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2004/UD,UM &UP=200482
```

```
et      Items   Description
S1      749655   (CACHE? OR   TEMPORARY()(STORAGE OR  MEMORY? )  OR  DISK? -
          OR BUFFER? OR RAM?)
S2     2226829   (WRIT??? OR WRIT???()DISK? OR TRANSFER??? OR RECORD??? OR -
          ENCOD??? OR UPDAT???)
S3       12118   (OLD OR ORIGINAL? OR PREVIOUS OR EARL??? OR OUTDAT???)(7N)-
          (NEW??? OR RECENT OR MODERN OR FRESH)
S4       75789   (MODIF???  OR MODIFICAT??? OR AMEND??? OR AMENDM??? OR CHA-
          NG??? OR ADJUST??? OR ADJUSTM???)(3N)(COP??? OR DATA OR FILE??
           OR INFORMATION?? OR INFO? OR VERSION? )
S5     1946417   (NODE? OR TERMINAL? OR COMPUTER? OR CLIENT? OR SERVER? OR -
          WORKSTATION?? OR STATION??)
S6       11093   (PERSIST??? OR PERSEVER??? OR ENDUR??? OR LINGER??? OR REM-
          AIN??? OR STEAD??? OR PROLONG??? OR UNCHANG???)(3N)(STOR??? OR
           STORAGE? OR MEMOR??? OR HARD()DRIVE OR DISK? OR CD()ROM?? OR
          CDROM OR ROM?? OR FLASH? OR EPROM?? OR PROM??)
S7      280446   (MASTER? OR CONTROL??? OR MAIN OR PRIMARY? OR SUPERVIS??? -
          OR ADMINISTRA??? OR MANAG?)(5N) S5
S8          37   AU=(CHANDRASEKARAN, S?  OR CHANDRASEKARAN S?)
S9          46   AU=(BAMFORD, R? OR BAMFORD R? )
S10         44   AU=(BRIDGE, W? OR  BRIDGE W?)
S11         38   AU=(BROWER, D? OR  BROWER D?)
S12          7   AU=(MACNAUGHTON, N?  OR MACNAUGHTON N?)
S13        294   AU=(CHAN, W? OR  CHAN W?)
S14          8   AU=(SRIHARI, V? OR SRIHARI V?)
S15        434   S8 OR S9 OR S10 OR S11 OR S12 OR S13 OR S14
S16          4   S8 AND S9 AND S10 AND S11 AND S12 AND S13 AND S14
S17          0   S15 AND S1 AND S2 AND S3
File 347:JAPIO Nov 1976-2004/Aug(Updated 041203)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2004/UD,UM &UP=200482
         (c) 2004  Thomson Derwent
```

?

16/3,K/1 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014845359 **Image available**
WPI Acc No: 2002-666065/200271
Related WPI Acc No: 1999-494629; 2002-339699; 2002-609207; 2002-636953
XRPX Acc No: N02-527002

Disk-write operation co-ordination method for database system, involves assigning higher priorities to data items corresponding to entries reacting latter one of two queues

Patent Assignee: ORACLE CORP (ORAC-N)
Inventor: **BAMFORD R J** ; **BRIDGE W H** ; **BROWER D** ; **CHAN W W S** ; **CHANDRASEKARAN S** ; **MACNAUGHTON N** ; **SRIHARI V**
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20020099729	A1	20020725	US 98199120	A	19981124	200271	B
			US 2001274270	P	20010307		
			US 200292047	A	20020304		

Priority Applications (No Type Date): US 2001274270 P 20010307; US 98199120 A 19981124; US 200292047 A 20020304
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20020099729	A1		32	G06F-012/00	CIP of application US 98199120
					Provisional application US 2001274270
					CIP of patent US 6353836

Inventor: **BAMFORD R J** ...

... **BRIDGE W H** ...

... **BROWER D** ...

... **CHAN W W S** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V**

16/3,K/2 (Item 2 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014845124 **Image available**
WPI Acc No: 2002-665830/200271
Related WPI Acc No: 1999-494629; 2002-609207; 2002-636953; 2002-666065
XRPX Acc No: N02-526783

Cache managing method for multiple cache system, involves providing master coordinating with multiple caches to write data item in persistent storage in response to data writing request

Patent Assignee: BAMFORD R J (BAMF-I); BRIDGE W H (BRID-I); BROWER D (BROW-I); CHAN W W S (CHAN-I); CHANDRASEKARAN S (CHAN-I); MACNAUGHTON N (MACN-I); SRIHARI V (SRIH-I)

Inventor: **BAMFORD R J ; BRIDGE W H ; BROWER D ; CHAN W W S ; CHANDRASEKARAN S ; MACNAUGHTON N ; SRIHARI V**
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20020095403	A1	20020718	US 98199120	A	19981124	200271	B
			US 2001274270	A	20010307		
			US 200291618	A	20020304		

Priority Applications (No Type Date): US 2001274270 P 20010307; US 98199120 A 19981124; US 200291618 A 20020304
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20020095403	A1		34	G06F-007/00	CIP of application US 98199120
					Provisional application US 2001274270
					CIP of patent US 6353836

Inventor: **BAMFORD R J** ...

... **BRIDGE W H** ...

... **BROWER D** ...

... **CHAN W W S** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V**

16/3,K/3 (Item 3 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014816247 **Image available**
WPI Acc No: 2002-636953/200268
Related WPI Acc No: 1999-494629; 2002-339699; 2002-609207; 2002-666065
XRPX Acc No: N02-503192
Cache management techniques for use in systems with multiple caches holding different copies of the same data items, uses a master to coordinate with the multiple caches
Patent Assignee: ORACLE INT CORP (ORAC-N); INT BUSINESS MACHINES CORP (IBMC)
Inventor: **BAMFORD R ; BRIDGE W ; BROWER D ; CHAN W ; CHANDRASEKARAN S ; MACNAUGHTON N ; SRIHARI V** ; AMINI L; FROSSARD P; VANDERGHEYNST P; VERSCHEURE O
Number of Countries: 101 Number of Patents: 005
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 200273416	A2	20020919	WO 2002US6981	A	20020307	200268	B
US 6628300	B2	20030930	US 2001334514	P	20011130	200367	
			US 2001334521	P	20011130		
			US 200292247	A	20020306		
EP 1412858	A2	20040428	EP 2002717580	A	20020307	200429	
			WO 2002US6981	A	20020307		
AU 2002248570	A1	20020924	AU 2002248570	A	20020307	200433	

CN 1524226   A   20040825 CN 2002806161   A   20020307 200477

Priority Applications (No Type Date): US 200292247 A 20020304; US 2001274270 P 20010307; US 2001334514 P 20011130; US 2001334521 P 20011130
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 200273416	A2	E	57	G06F-012/00	

Designated States (National): AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ OM PH PL PT RO RU SD SE SG SI SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA ZM ZW
Designated States (Regional): AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZM ZW

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6628300	B2			G09G-005/00	Provisional application US 2001334514
					Provisional application US 2001334521
EP 1412858	A2	E		G06F-012/00	Based on patent WO 200273416

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
AU 2002248570	A1			G06F-012/00	Based on patent WO 200273416
CN 1524226	A			G06F-012/00	

Inventor: **BAMFORD R** ...

... **BRIDGE W** ...

... **BROWER D** ...

... **CHAN W** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V**

16/3,K/4   (Item 4 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014788501   **Image available**
WPI Acc No: 2002-609207/200265
Related WPI Acc No: 1999-494629; 2002-339699; 2002-636953; 2002-666065
XRPX Acc No: N02-482415

Managing system caches by sending write notification message from master to node indicating persistent storage of most recent modified data item version

Patent Assignee: ORACLE INT CORP (ORAC-N); ORACLE CORP (ORAC-N); BAMFORD R J (BAMF-I); BRIDGE W H (BRID-I); BROWER D (BROW-I); CHAN W W S (CHAN-I); CHANDRASEKARAN S (CHAN-I); MACNAUGHTON N (MACN-I); SRIHARI V (SRIH-I)
Inventor: **BAMFORD R** ; **BRIDGE W** ; **BROWER D** ; **CHAN W** ; **CHANDRASEKARAN S** ; **MACNAUGHTON N** ; **SRIHARI V** ; **BAMFORD R J** ; **BRIDGE W H** ; **CHAN W W S**
Number of Countries: 101 Number of Patents: 006
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 200271229	A2	20020912	WO 2002US7475	A	20020306	200265	B
US 20020095403	A1	20020718	US 98199120	A	19981124	200271	
			US 2001274270	P	20010307		

```
                              US 200291618    A   20020304
EP 1366420     A2  20031203  EP 2002748408   A   20020306  200380
                              WO 2002US7475   A   20020306
AU 2002248570  A1  20020924  AU 2002248570   A   20020307  200433
AU 2002335503  A1  20020919  AU 2002335503   A   20020306  200433
CN 1496510     A   20040512  CN 2002806162   A   20020306  200452
                              WO 2002US7475   A   20020306

Priority Applications (No Type Date): US 200291618 A 20020304; US
  2001274270 P 20010307; US 98199120 A 19981124; US 200292247 A 20020304
Patent Details:
Patent No  Kind Lan Pg   Main IPC    Filing Notes
WO 200271229  A2 E  64 G06F-012/00
   Designated States (National): AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA
   CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN
   IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ
   OM PH PL PT RO RU SD SE SG SI SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA
   ZM ZW
   Designated States (Regional): AT BE CH CY DE DK EA ES FI FR GB GH GM GR
   IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZM ZW
US 20020095403 A1    34 G06F-007/00   CIP of application US 98199120
                                     Provisional application US 2001274270
                                     CIP of patent US 6353836
EP 1366420    A2 E     G06F-012/08   Based on patent WO 200271229
   Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT
   LI LT LU LV MC MK NL PT RO SE SI TR
AU 2002248570 A1       G06F-012/00   Based on patent WO 200273416
AU 2002335503 A1       G06F-012/00   Based on patent WO 200271229
CN 1496510    A        G06F-012/08
```

Inventor: **BAMFORD R** ...

... **BRIDGE W** ...

... **BROWER D** ...

... **CHAN W** ...

... **CHANDRASEKARAN S** ...

... **MACNAUGHTON N** ...

... **SRIHARI V** ...

... **BAMFORD R J** ...

... **BRIDGE W H** ...
... **CHAN W W S**
?

42/3,K/1 (Item 1 from file: 347)
DIALOG(R)File 347:JAPIO

03864242 **Image available**
TAKEOVER MECHANISM FOR TROUBLE LOG INFORMATION OF COMPUTER SYSTEM

PUB. NO.: 04-229342 [JP 4229342 A]
PUBLISHED: August 18, 1992 (19920818)
INVENTOR(s): OKAMOTO AKIRA
FURUSAWA KAZUKO
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
NEC SOFTWARE LTD [491061] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 02-414880 [JP 90414880]
FILED: December 27, 1990 (19901227)
JOURNAL: Section: P, Section No. 1461, Vol. 16, No. 580, Pg. 104, December 18, 1992 (19921218)

INTL CLASS: **G06F-011/34 ; G06F-012/00**

ABSTRACT

PURPOSE: To make next-time operation take over the trouble log information **remaining** on a **main** **storage** when the **computer** system stops...

42/3,K/2 (Item 2 from file: 347)
DIALOG(R)File 347:JAPIO

03797075 **Image available**
DOCUMENT REVISION SUPPORT SYSTEM

PUB. NO.: 04-162175 [JP 4162175 A]
PUBLISHED: June 05, 1992 (19920605)
INVENTOR(s): AKAOKA NAOHITO
APPLICANT(s): TOSHIBA CORP [000307] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 02-287272 [JP 90287272]
FILED: October 26, 1990 (19901026)
JOURNAL: Section: P, Section No. 1426, Vol. 16, No. 457, Pg. 117, September 22, 1992 (19920922)

INTL CLASS: **G06F-015/20**

ABSTRACT

... output device 1, the revised part is designated and the revised content is inputted. The **computer** mainbody 2 generates a revised present specification collection file and revised content file in a **disk** device 3. This is repeated and the revised present specification collection file and the revised content file are sequentially **updated** . **New / old** collation table data is generated and printed out. A present specification collection file is rewritten...

42/3,K/3 (Item 3 from file: 347)
DIALOG(R)File 347:JAPIO

02748560 **Image available**
DATA TRANSMISSION SYSTEM TO CENTER IN DECENTRALIZED PROCESSING SYSTEM

PUB. NO.: 01-046160 [JP 1046160 A]
PUBLISHED: February 20, 1989 (19890220)
INVENTOR(s): MATSUBARA MOICHI
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 62-202951 [JP 87202951]
FILED: August 14, 1987 (19870814)
JOURNAL: Section: P, Section No. 880, Vol. 13, No. 242, Pg. 134, June 07, 1989 (19890607)

INTL CLASS: **G06F-015/21**

ABSTRACT
PURPOSE: To continue the input and processing of data received from a **terminal** equipment in a **terminal controller** by sending untransmitted data if **remaining** in a data **storing** device to the center in each prescribed cycle in parallel with a job to supply...

42/3,K/4 (Item 4 from file: 347)
DIALOG(R)File 347:JAPIO

02519266 **Image available**
DATA PROCESSING SYSTEM FOR DIAGNOSIS/TREATMENT POINT

PUB. NO.: 63-136166 [JP 63136166 A]
PUBLISHED: June 08, 1988 (19880608)
INVENTOR(s): ITAYA TOYOYUKI
HAGIO SHIGEJI
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
NEC SOFTWARE LTD [491061] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 61-283335 [JP 86283335]
FILED: November 27, 1986 (19861127)
JOURNAL: Section: P, Section No. 773, Vol. 12, No. 394, Pg. 164, October 20, 1988 (19881020)

INTL CLASS: **G06F-015/21**

ABSTRACT
... A diagnosis/treatment information file 51 containing data equivalent to one month or several months **recorded** on a magnetic tape, etc., is prepared together with a **new / old** point master file 52 containing at least two latest diagnosis/treatment point lists **written** in a magnetic **disk** , etc. The file 51 is read into a core memory in a **computer** unit 3 and the **new / old** points are calculated for each series of diagnosis/treatment actions. Then a new/old comparison...

42/3,K/5 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

012907865 **Image available**
WPI Acc No: 2000-079701/200007
XRPX Acc No: N00-062966

Data server for data communication services such as electronic mail service etc., - outputs excess warning and deletion warning when data addressed to particular terminal assigned with specific memory capacity exceeds allotted capacity, and deletes portion of data

Patent Assignee: SONY CORP (SONY)
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
JP 11328059	A	19991130	JP 98133309	A	19980515	200007	B

Priority Applications (No Type Date): JP 98133309 A 19980515
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
JP 11328059	A		30	G06F-013/00	

...Abstract (Basic): ADVANTAGE - Enables effective storage **management** process. Informs each **terminal** about **remaining** **storage** capacity, so that future information receiving process can be planned. Presents excess memory usage capacity...

International Patent Class (Main): **G06F-013/00**

42/3,K/6 (Item 2 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011931075 **Image available**
WPI Acc No: 1998-347985/199830
Related WPI Acc No: 1999-633513
XRPX Acc No: N98-271675

Persistent object management method for OO database - involves creating persistent objects encapsulating logical data units which are translated to OO types

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: BLACKMAN K R; HOWE J L
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5765162	A	19980609	US 96736952	A	19961025	199830	B

Priority Applications (No Type Date): US 96736952 A 19961025
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5765162	A		11	G06F-017/30	

...Abstract (Basic): computer for organizing the datastore persistent objects. Tracking the datastore collections, schema mappers and datastore **persistent** objects in the **memory** of the **computer** using an instance **manager** .

International Patent Class (Main): **G06F-017/30**

42/3,K/7 (Item 3 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011930978 **Image available**
WPI Acc No: 1998-347888/199830
XRPX Acc No: N98-271578
Multi-node database segment flushing method - involves defining persistent storage **semaphore for database segments accessible by** primary **and backup** nodes
Patent Assignee: NCR CORP (NATC)
Inventor: CATOZZI J R; RABINOVICI S
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5764905	A	19980609	US 96711235	A	19960909	199830	B

Priority Applications (No Type Date): US 96711235 A 19960909
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5764905	A		10	G06F-015/163	

... **involves defining** persistent storage **semaphore for database segments accessible by** primary **and backup** nodes

...Abstract (Basic): accessed database segments, and storing a copy of the modified database segments in both a **primary node** and a backup **node** . A **persistent storage** semaphore is defined for the database segments on the data **storage** device, the **persistent storage** semaphore being accessible by both the **primary** and backup **nodes** .

International Patent Class (Main): **G06F-015/163**

42/3,K/8 (Item 4 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011586056 **Image available**
WPI Acc No: 1998-003185/199801
XRPX Acc No: N98-002331
Game system for pachinko, slot machine - has main computer which calculates the remaining number of coins in memory after supply of coins
Patent Assignee: SOGO JOHO KIKI HANBAI KK (SOGO-N)
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
JP 9271574	A	19971021	JP 9726589	A	19970210	199801	B

Priority Applications (No Type Date): JP 9624474 A 19960209
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
JP 9271574	A		12	A63F-007/02	

...Abstract (Basic): coin from coin issue machine (20) is approved only when recitation numbers are coinciding. A **main computer** calculates the number of coins **remaining** in the **memory** after each batch of coin supply...
International Patent Class (Additional): **G06F-017/60** ...

42/3,K/9 (Item 5 from file: 350)

DIALOG(R)File 350:Derwent WPIX

010608568 **Image available**
WPI Acc No: 1996-105521/199611
XRPX Acc No: N96-088470

Microprocessor system with RAID disk drive storage - provides non-volatile RAM with new data, copy of old data, copy of old parity, and sync. state indicator, determines new parity and transmits new data to data node for storage, and scans RAM for non-reset state indicator at power-up after p

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: MENDELSOHN N
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5488731	A	19960130	US 92924219	A	19920803	199611	B
			US 94352428	A	19941208		

Priority Applications (No Type Date): US 92924219 A 19920803; US 94352428 A 19941208
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5488731	A		11	G06F-011/00	Cont of application US 92924219

...Abstract (Basic): The multiprocessor system includes identical nodes interconnected through a switching network, each **node** including a disk drive, Non-volatile RAM, and a processor. The system stores data in either a RAID or mirrored fashion across **disk** drives in different **nodes** . When data is stored in a RAID, an **RAM** in a parity **node** is provided with an entry including the **new** data, a copy of **old** data from the **node** to which the **new** data is to be **written** , a copy of the old parity, and a synchronization state indicator...

International Patent Class (Main): **G06F-011/00**
?

32/3,K/1 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

012465742 **Image available**
WPI Acc No: 1999-271850/199923
XRPX Acc No: N99-203415
Data transferring method for information processing system such as general purpose computer - involves reading data management information from old disk **controller and** writes **in** new disk **controller in intrinsic logic address via data** transfer **path**
Patent Assignee: HITACHI LTD (HITA)
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
JP 11085576	A	19990330	JP 97239525	A	19970904	199923	B

Priority Applications (No Type Date): JP 97239525 A 19970904
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
JP 11085576	A		14	G06F-012/00	

... **involves reading data management information from** old disk **controller and** writes **in** new disk **controller in intrinsic logic address via data** transfer **path**
...Abstract (Basic): NOVELTY - Based on read data management information (111), a **new disk** controller (102) reads out data in **old disk** controller (104), partially and **writes** in the new **disk** controller in intrinsic logic address via a data **transfer** path (113). DETAILED DESCRIPTION - CPU (101) and new **disk** controller (102) are connected to a new **disk** potentiometer (103) by currently **changing** higher order **data transfer** path (114) connected between the old **disk** controller (104) and old **disk** potentiometer (105). Data **transfer** path (113) is setup between the controllers (102,104). Data management information (111) in the old **disk** controller (104) is read-out from it to new **disk** controller (102) via a data **transfer** path (113...

...ADVANTAGE - Since smooth data **transfer** is performed, efficient and various data transferring is achieved. DESCRIPTION OF DRAWING(S) - The figure shows block diagram of information processing system. (102) **New disk** controller; (103) **New disk** potentiometer; (104) **Old disk** controller; (105) **Old disk** potentiometer; (113) Data **transfer** path; (114) Higher order data **transfer** path...
...Title Terms: **TRANSFER** ;
International Patent Class (Main): **G06F-012/00**
International Patent Class (Additional): **G06F-003/06**

32/3,K/2 (Item 2 from file: 350)
DIALOG(R)File 350:Derwent WPIX

012253492 **Image available**
WPI Acc No: 1999-059599/199905
Related WPI Acc No: 1993-303701; 1999-328972; 2002-187997; 2003-057318
XRPX Acc No: N99-044416
Computer **jukebox** management **system - has processor which displays one of advertisement on screen in response to receipt of data representing**

number of times advertisement to be run, when jukebox does not generate selected song signal
Patent Assignee: ARACHNID INC (ARAC-N)
Inventor: MARTIN J R; TILLERY M L; ZAMMUTO S N
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5848398	A	19981208	US 90538981	A	19900615	199905	B
			US 94268782	A	19940630		
			US 96584253	A	19960111		
			US 96638022	A	19960425		

Priority Applications (No Type Date): US 96638022 A 19960425; US 90538981 A 19900615; US 94268782 A 19940630; US 96584253 A 19960111
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5848398	A		11	G06F-017/60	Cont of application US 90538981
					Cont of application US 94268782
					CIP of application US 96584253
					CIP of patent US 5781889

Computer **jukebox** management **system**...

...Abstract (Basic): ADVANTAGE - Simplifies loading and erasing of **old** or **new** recordings into or from memory of each computer jukebox respectively. Prevents theft of **recording disk** due to utilisation of memory that stores digitised song data. Enhances system to be used ...
International Patent Class (Main): **G06F-017/60**

32/3,K/3 (Item 3 from file: 350)
DIALOG(R)File 350:Derwent WPIX

012192961 **Image available**
WPI Acc No: 1998-609874/199851
XRPX Acc No: N98-474463
Difference file generating method for personal computer system - involves searching old **file data string matching with** new **file data strings based on which indication of position of matching string in** old **file or** new **file data string is stored in difference file**
Patent Assignee: MILLER CALL PLAUCK & MILLER (MILL-N)
Inventor: MILLER W A
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5832520	A	19981103	US 9621457	A	19960703	199851	B
			US 96754486	A	19961122		

Priority Applications (No Type Date): US 9621457 P 19960703; US 96754486 A 19961122
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5832520	A		25	G06F-017/30	Provisional application US 9621457

... **involves searching** old **file data string matching with** new **file data strings based on which indication of position of matching string in** old **file or** new **file data string is stored in difference file**

...Abstract (Basic): The method involves reading new file data strings from a **new** file. An **old** file is searched for the presence of **old** file data strings matching with the **new** file data strings. An indication of the position of the matching strings in the old...

...in the new file has been read and corresponding insert and copy operations have been **written** into the difference file...

...ADVANTAGE - Creates copy of **new** file. Quickly using duplicate of **old** file. Builds difference file in multi-step. Process to minimise size of difference **file** . **Adjusts** memory size required for hash table allowing improved search speed...

...Title Terms: **DISK ;**

International Patent Class (Main): **G06F-017/30**

32/3,K/4 (Item 4 from file: 350)

DIALOG(R)File 350:Derwent WPIX

011601583 **Image available**

WPI Acc No: 1998-018711/199802

XRPX Acc No: N98-014195

Redundant media device mass storage system - has redundant array media adaptor cards under real time operating system control, which provides targetted reception of data broadcast from SCSI hosts to data bus reading devices

Patent Assignee: STORAGE COMPUTER CORP (STOR-N)

Inventor: SARKOZY A; VALENTINO J

Number of Countries: 023 Number of Patents: 003

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 9744735	A1	19971127	WO 97US8253	A	19970515	199802	B
US 5720027	A	19980217	US 96652636	A	19960521	199814	
AU 9731271	A	19971209	AU 9731271	A	19970515	199824	

Priority Applications (No Type Date): US 96652636 A 19960521

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 9744735	A1	E	53	G06F-011/34	
Designated States (National): AU CA CN JP KR					
Designated States (Regional): AT BE CH DE DK ES FI FR GB GR IE IT LU MC NL PT SE					
US 5720027	A		18	G06F-011/08	
AU 9731271	A			G06F-011/34	Based on patent WO 9744735

...Abstract (Basic): the data bus (72), e.g. computer central memory (60) and a number of storage **disks** (parity (50) and data (55)), under control of a real time operating system. The redundant array **computer** operating system (100) provides the **control** and selected designation of the disc adapters (95) as targeted receivers to read data that is 'broadcast' over the data bus, providing simultaneous **transfer** of data over the data bus (72...

...adapter further includes exclusive-OR logic (93A) to provide direct calculation of parity from the **newly** received data and a subsequently received **old** data on a single subsequent data bus cycle...

...ADVANTAGE - Allows for simultaneous **transfers** over XBUS, and allows hardware parity calculation on triple adapter board...
International Patent Class (Main): **G06F-011/08** ...

... **G06F-011/34**
International Patent Class (Additional): **G06F-011/00** ...

... **G06F-011/14** ...

... **G06F-011/16** ...

... **G06F-011/22**

32/3,K/5 (Item 5 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011388215 **Image available**
WPI Acc No: 1997-366122/199734
XRPX Acc No: N97-304271
Method for storing data in fault tolerant storage device with disk **units - identifies array target areas to** write new **data, one area has** old **data and is associated with stripes to define** updated **stripe, in absence of read step** writes **new data to target area defining unprotected** data **, monitors and** adjusts **it**
Patent Assignee: HEWLETT-PACKARD CO (HEWP)
Inventor: SAVAGE S; WILKES A J
Number of Countries: 005 Number of Patents: 003
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 785512	A2	19970723	EP 96118133	A	19961112	199734	B
JP 9231017	A	19970905	JP 972533	A	19970110	199746	
US 5720025	A	19980217	US 96588140	A	19960118	199814	

Priority Applications (No Type Date): US 96588140 A 19960118
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 785512	A2	E	13	G06F-011/10	
Designated States (Regional): DE FR GB					
JP 9231017	A		13	G06F-003/06	
US 5720025	A		12	G06F-011/14	

Method for storing data in fault tolerant storage device with disk **units...**

...**identifies array target** areas to write new **data, one area has** old **data and is associated** with stripes to define updated **stripe, in absence of read step** writes **new data to target area defining unprotected** data **, monitors and** adjusts **it**

...Abstract (Basic): The method stores data in a fault tolerant storage device having several **disk** unit (18 to 26) forming a **disk** array (16) with several stripes (28) for storing data and parity information. The method identifies one or more target areas (30 to 38) on the array to **write** **new** data where one area has **old** data and is associated with the stripes to define an **updated** stripe...

...In the absence of a reading step, new data is **written** to the target area defining unprotected data. A quantity of the unprotected data present on...

...systems for computers, and to fault tolerant storage architecture suited for redundant Arrays of Independent **Disks** (RAID...

...ADVANTAGE - Architecture relaxes coherency between parity and data to reduce access time necessary to **update** data stored on array of **disks** .

...Abstract (Equivalent): The method stores data in a fault tolerant storage device having several **disk** unit (18 to 26) forming a **disk** array (16) with several stripes (28) for storing data and parity information. The method identifies one or more target areas (30 to 38) on the array to **write** **new** data where one area has **old** data and is associated with the stripes to define an **updated** stripe...

...In the absence of a reading step, new data is **written** to the target area defining unprotected data. A quantity of the unprotected data present on...

...systems for computers, and to fault tolerant storage architecture suited for redundant Arrays of Independent **Disks** (RAID...

...ADVANTAGE - Architecture relaxes coherency between parity and data to reduce access time necessary to **update** data stored on array of **disks** .

...Title Terms: **WRITING** ;
International Patent Class (Main): **G06F-003/06** ...

... **G06F-011/10** ...

... **G06F-011/14**
International Patent Class (Additional): **G06F-012/16**

32/3,K/6 (Item 6 from file: 350)
DIALOG(R)File 350:Derwent WPIX

011121890 **Image available**
WPI Acc No: 1997-099815/199709
XRPX Acc No: N97-082615
Method of XORing in disk **controller of** disk **drive - involves generating block of new parity information by XORing block of** new **data, block of** old **data and block of old parity information**
Patent Assignee: EMC CORP (EMCE-N)
Inventor: WILCOX J A; WINKLER J L
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5594862	A	19970114	US 94278870	A	19940720	199709	B

Priority Applications (No Type Date): US 94278870 A 19940720
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5594862	A		18	G06F-011/00	

Method of XORing in disk **controller of** disk **drive...**

...**involves generating block of new parity information by XORing block of** new **data, block of** old **data and block of old parity information**

...Abstract (Basic): The method involves receiving in a **disk** **controller** from a host **computer** at least one block of new data to be **written** to **disk** drives commencing at a first **disk** address. A block of new data are stored commencing at a first one of the **cache** memory locations A first control signal is provided from a microprocessor within the **disk** controller to an XOR calculation controller. A block of new parity information is generated by XORing the block of **new** data, the block of **old** data and the block of old parity information in response to receipt by the XOR...

...amount of time storage subsystem microprocessor uses to perform required XOR operations on data being **transferred** from host...
International Patent Class (Main): **G06F-011/00**

32/3,K/7 (Item 7 from file: 350)
DIALOG(R)File 350:Derwent WPIX

010476559 **Image available**
WPI Acc No: 1995-377880/199549
Program remote revision system for computers in remote place - transmits new edition operating system from remote revision management computer **to system volume for new edition of computer for revision**
Patent Assignee: HITACHI LTD (HITA); HITACHI SOFTWARE ENG CO LTD (HISF)
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
JP 7253891	A	19951003	JP 9444128	A	19940315	199549	B

Priority Applications (No Type Date): JP 9444128 A 19940315
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
JP 7253891	A		5	G06F-009/445	

... **transmits new edition operating system from remote revision** management computer **to system volume for new edition of computer for revision**

...Abstract (Basic): The program remote revision system has a remote revision **management** **computer** (2) which revises the operating system of a computer (4) in a remote place. The computer to be revised and the remote revision **management** **computer** are connected through a connection network (10). The remote revision **management** **computer** transmits the revision program stored in a **disk** (3) to a program revision work **disk** (7) through the connection network, based on the revision indication from a terminal equipment (1). The revision program received by the work **disk** is stored in a volume for **new** edition (9). The **old** version under operation is stored in a system volume in operation (8). The new version address is **written** in a system starting volume address storage memory (5). The computer is **changed** to the new **version** at the time of system recovery...
International Patent Class (Main): **G06F-009/445**

International Patent Class (Additional): **G06F-009/06**

32/3,K/8 (Item 8 from file: 350)
DIALOG(R)File 350:Derwent WPIX

010418924 **Image available**
WPI Acc No: 1995-320239/199541
Related WPI Acc No: 1994-176617; 1996-251402
XRPX Acc No: N95-240934
Computerised office automation system with audit history - has unit creating chronologically ordered historical data image records **with new data added for every** update **to give** record **history**
Patent Assignee: CISGEM TECHNOLOGIES INC (CISG-N)
Inventor: MURDOCK D M
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 5448729	A	19950905	US 90471290	A	19900126	199541	B
			US 92954416	A	19920930		
			US 94249883	A	19940525		

Priority Applications (No Type Date): US 90471290 A 19900126; US 92954416 A 19920930; US 94249883 A 19940525
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5448729	A		84	G06F-017/30	Cont of application US 90471290
					Cont of application US 92954416
					Cont of patent US 5317733

... **has unit creating chronologically ordered historical data image** records **with new data added for every** update **to give** record **history**

...Abstract (Basic): The system has a computerised database with **updating** input facility. An audit history of the database **records** is maintained using a unit retrieving original active and delta **records** . A unit creates a first chronologically ordered historical data image **record** in **temporary storage** . This **record** includes **new** data associated with the **record** after **original** data has been **updated** and stored in a first session...

...Each subsequent delta **record** is applied to the historical data image until exhausted. Each **record** then represents data corresponding to the **record** each time it was stored into the database. Each **record** has an associated active **record** and at least one delta **record** including **changed record data** ID and a location of the **changed data** . The historical **data** image **records** can be displayed...
...Title Terms: **RECORD ;**
International Patent Class (Main): **G06F-017/30**

32/3,K/9 (Item 9 from file: 350)
DIALOG(R)File 350:Derwent WPIX

010136748 **Image available**
WPI Acc No: 1995-037999/199506
Related WPI Acc No: 1992-278088; 1992-425865

XRPX Acc No: N95-030061

Encoding **and rebuilding of data contents of up to two unavailable DASDs - by generating (M-1)xM data block from (M-1)x(M-2) block array to enable erased or unavailable portion of data array to be rebuilt**

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)

Inventor: BLAUM M M; BRADY J; BRUCK J; MENON J M; BRADY J T

Number of Countries: 004 Number of Patents: 005

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 632376	A2	19950104	EP 94109687	A	19940623	199506	B
EP 632376	A3	19950301				199541	
US 5579475	A	19961126	US 91653596	A	19910211	199702	
			US 91718724	A	19910621		
			US 9385707	A	19930630		
			US 95518955	A	19950824		
EP 632376	B1	19980211	EP 94109687	A	19940623	199811	
DE 69408498	E	19980319	DE 608498	A	19940623	199817	
			EP 94109687	A	19940623		

Priority Applications (No Type Date): US 9385707 A 19930630; US 91653596 A 19910211; US 91718724 A 19910621; US 95518955 A 19950824

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 632376	A2	E	20	G06F-011/00	
Designated States (Regional): DE FR GB					
US 5579475	A		16	G06F-011/34	CIP of application US 91653596
					CIP of application US 91718724
					Cont of application US 9385707
					CIP of patent US 5271012
EP 632376	B1	E	23	G06F-011/00	
Designated States (Regional): DE FR GB					
DE 69408498	E			G06F-011/00	Based on patent EP 632376
EP 632376	A3			G06F-011/00	

Encoding **and rebuilding of data contents of up to two unavailable DASDs** ...

...Abstract (Basic): The blocks in each row of the (M-1)*M data array are **written** to counterpart ones of the M failure independent of direct access storage devices (DASDs). If...

...than (M-2) available DASDs on a scheduled or opportunistic basis. The rebuilt portion is **written** either to counterpart spare DASDs or to spare space available on no less than M...

...Abstract (Equivalent): The blocks in each row of the (M-1)*M data array are **written** to counterpart ones of the M failure independent of direct access storage devices (DASDs). If...

...than (M-2) available DASDs on a scheduled or opportunistic basis. The rebuilt portion is **written** either to counterpart spare DASDs or to spare space available on no less than M...

...Abstract (Equivalent): an integer number M of said plurality in a higher level redundant array of inexpensive **disks** (RAID) configuration, M being a prime number, said accessing means being responsive to a source of external commands for reading and **writing** of data blocks stored on selected ones of said M of said plurality of DASDs...

...2) data blocks in each of the (M-1) rows in said logical array being **written** across counterpart ones of the M-2 of the DASDs, ones of said

plurality being...

...c) **writing** the generated (M-1)*M logical array across the M DASDs...

...d) responsive to a **write update** command to **modify** a **data** block NOT located on the major diagonal of the logical array by reading the old data block, an old diagonal parity value, and an **old** row parity value, calculating a **new** diagonal and a new row parity value, and **writing** the **modified data** block and the new diagonal and row parity values in place among the M DASDs, and responsive to a **write update** command to **modify** a **data** block located on the major diagonal, repeating steps (a) and (b) with respect to generation of (M-1) new diagonal parity values and the new row parity value and **writing** the **modified data** block and new diagonal parities and new row parity in place among the M DASDs...

...a portion of said logical array from not less than (M-2) available DASDs and **writing** said rebuilt unavailable portion to counterpart ones of said plurality of DASDs operatively designated as...

Title Terms: **ENCODE ;**

International Patent Class (Main): **G06F-011/00** ...

... **G06F-011/34**

International Patent Class (Additional): **G06F-011/10**

32/3,K/10 (Item 10 from file: 350)

DIALOG(R)File 350:Derwent WPIX

009580891 **Image available**

WPI Acc No: 1993-274437/199335

XRPX Acc No: N93-210746

EEPROM data rewriting method for EEPROM card with SRAM - reading data to be rewritten out of EEPROM into RAM **where it is** changed **and reading new** data **out of** RAM **for** writing **into EEPROM and comparing data read out from** RAM **and rewritten data read out of EEPROM**

Patent Assignee: FUJI PHOTO FILM CO LTD (FUJF)

Inventor: SAITO O

Number of Countries: 004 Number of Patents: 005

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 557968	A2	19930901	EP 93102875	A	19930224	199335	B
US 5390148	A	19950214	US 9320541	A	19930222	199512	
EP 557968	A3	19940330	EP 93102875	A	19930224	199521	
EP 557968	B1	19990901	EP 93102875	A	19930224	199940	
DE 69326175	E	19991007	DE 626175	A	19930224	199947	
			EP 93102875	A	19930224		

Priority Applications (No Type Date): JP 9273255 A 19920225

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 557968	A2	E	15	G11C-016/06	
Designated States (Regional): DE FR GB					
US 5390148	A		14	G11C-016/00	
EP 557968	B1	E		G11C-016/06	
Designated States (Regional): DE FR GB					
DE 69326175	E			G11C-016/06	Based on patent EP 557968

EP 557968 A3 G11C-016/06

... **reading data to be rewritten out of EEPROM into** RAM **where it is** changed **and reading new** data **out of** RAM **for** writing **into EEPROM and comparing data read out from** RAM **and rewritten data read out of EEPROM**

...Abstract (Basic): old data that is to be rewritten and reading out this data, which is then **written** to an SRAM where it is temporarily stored. The **old** data in the SRAM is rewritten to **new** data, which is read out of the SRAM and **written** to the EEPROM pref. by addressing the same EEPROM area that stored the old data...

...A comparator determines whether all bits of the new data rewritten in the SRAM and **written** in the EEPROM coincide, verifying the **changed data** .
...

...g. electronic still video image camera memory card. Improved reliability; prevents data loss due to **write** failure
...Abstract (Equivalent): been stored in the first area of the EEPROM. The first unit of data is **written** , which has been read out of the first area of the EEPROM, into a **RAM** , and a second unit of data is generated in the **RAM** by rewriting **new** data for the **old** data into the first unit of data that has been **written** in the **RAM** .
...

...in which the second unit of data, which includes the new data, is to be **written** , and the second unit of data is **written** , which has been read out of the **RAM** , into the second area of the EEPROM. The second unit of data is read out of the **RAM** , and the second unit of data is read out of the second area of the
...Title Terms: **RAM** ;
International Patent Class (Additional): **G06F-012/16** ...

32/3,K/11 (Item 11 from file: 350)
DIALOG(R)File 350:Derwent WPIX

009513756
WPI Acc No: 1993-207292/199326
XRPX Acc No: N93-159449
Personal computer **memory** controller **- pipelines successive snoop cycles having a basic snoop cycle duration longer than memory** write **cycle duration**
Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: DERWIN M T; WALL W A
Number of Countries: 004 Number of Patents: 007
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 549164	A1	19930630	EP 92311061	A	19921203	199326	B
BR 9204884	A	19930622	BR 924884	A	19921207	199329	
CA 2077048	A	19930621	CA 2077048	A	19920827	199337	
US 5341487	A	19940823	US 91812196	A	19911220	199433	
EP 549164	B1	19950802	EP 92311061	A	19921203	199535	
DE 69203842	E	19950907	DE 603842	A	19921203	199541	

			EP 92311061	A	19921203	
CA 2077048	C	19961022	CA 2077048	A	19920827	199702

Priority Applications (No Type Date): US 91812196 A 19911220

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 549164	A1	E	10	G06F-012/08	
US 5341487	A		9	G06F-012/00	
EP 549164	B1	E	14	G06F-012/08	
DE 69203842	E			G06F-012/08	Based on patent EP 549164
BR 9204884	A			G06F-015/40	
CA 2077048	A			G06F-013/16	
CA 2077048	C			G06F-013/16	

Personal computer **memory** controller - ...

...**pipelines** successive snoop cycles having a basic snoop cycle duration **longer than memory** write **cycle duration**

...Abstract (Basic): The memory controller includes a snoop mechanism operable during the memory **write** cycle to determine whether a snoop cycle is required. If a snoop cycle is required the snoop mechanism generates control signals for controlling invalidation cycles in the **cache** to invalidate the **cache** line corresponding to the planar memory (16) address **written** to by the bus master (52, 54...

...Monitoring logic monitors control signals from the bus master to determine whether a second memory **write** cycle has been started before a basic snoop cycle has finished and generates a corresponding control signal. The snoop mechanism initiates a snoop cycle of same duration as a memory **write** cycle immediately after the basic snoop cycle, on receipt of the control signal...

...USE/ADVANTAGE - For computer system having a memory with a **write** -through **cache** accessible by more than one device. Allows a bus master to run at full speed to perform successive **write** cycles

...Abstract (Equivalent): Memory **controller** for a **computer** having a memory system including a planar memory (16) and a **write** -through **cache** (68), and bus master means (52, 54) being capable of making a plurality of planar **writes** during consecutive memory **write** cycles which **writes** require invalidating different **cache** lines, the memory controller being connectable to said memory system and said bus master means...

...consecutive memory cycles, to cause a basic snoop cycle to occur to invalidate a first **cache** line corresponding to a planar memory location being **written** to in said first memory cycle, said basic snoop cycle being longer than a memory **write** cycle; characterised by the snoop mechanism comprising monitoring logic for monitoring control signals from said bus master means (52, 54) to determined if a second memory **write** cycle has been started before said basic snoop cycle has finished and generating a control...

...snoop cycle immediately after and pipelined with said basic snoop cycle, to invalidate a second **cache** line in response to receiving said control signal, said pipeline snoop cycle having the same duration as the memory **write** cycle

...Abstract (Equivalent): The personal computer has a memory system

including a **write** -through **cache** which is accessible by more than one device. A snoop mechanism includes logic that monitors bus master control signals to determine if a new memory **write** cycle has been started before a current snoop cycle has finished...

...a new cycle ha been started, then a corresponding snoop cycle occurs which overlaps the **new** memory cycle and is pipelined with the **previous** snoop cycle so that the snooping mechanism does not fall behind the memory **write** cycles...

...ADVANTAGE - Snoop cycle determines before it ends that another memory **write** cycle is occurring which requires subsequent pipeline snoop cycle
...Title Terms: **WRITING** ;
International Patent Class (Main): **G06F-012/08** ...

... **G06F-013/16** ...

... **G06F-015/40**
International Patent Class (Additional): **G06F-009/38**

32/3,K/12 (Item 12 from file: 350)
DIALOG(R)File 350:Derwent WPIX

009324831 **Image available**
WPI Acc No: 1993-018295/199302
XRPX Acc No: N93-013934
Computer performance improvements using simulated cache **associativity - simulates effect of set associative** cache **by detecting** cache **misses and re-mapping pages on main memory**
Patent Assignee: DIGITAL EQUIP CORP (DIGI)
Inventor: SITES R L
Number of Countries: 037 Number of Patents: 012
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 9222867	A1	19921223	WO 92US4281	A	19920521	199302	B
AU 9222474	A	19930112	AU 9222474	A	19920521	199317	
EP 543991	A1	19930602	EP 92914461	A	19920521	199322	
			WO 92US4281	A	19920521		
JP 5509189	W	19931216	WO 92US4281	A	19920521	199404	
			JP 93500873	A	19920521		
TW 219986	A	19940201	TW 91107803	A	19911003	199413	
AU 658914	B	19950504	AU 9222474	A	19920521	199526	
US 5442571	A	19950815	US 91716397	A	19910617	199538	
			US 94250315	A	19940527		
IL 102001	A	19960131	IL 102001	A	19920526	199617	
CA 2088779	C	19980901	CA 2088779	A	19920521	199845	
KR 9605443	B1	19960425	WO 92US4281	A	19920521	199915	
			KR 93700415	A	19930213		
EP 543991	B1	19990728	EP 92914461	A	19920521	199934	
			WO 92US4281	A	19920521		
DE 69229667	E	19990902	DE 629667	A	19920521	199942	
			EP 92914461	A	19920521		
			WO 92US4281	A	19920521		

Priority Applications (No Type Date): US 91716397 A 19910617; US 94250315 A

19940527

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 9222867	A1	E	16	G06F-012/08	
Designated States (National): AT AU BB BG BR CA CH DE DK ES FI GB HU JP KP KR LK LU MG MW NL NO PL RO RU SD SE					
Designated States (Regional): AT BE CH DE DK ES FR GB GR IT LU MC NL OA SE					
AU 9222474	A			G06F-012/08	Based on patent WO 9222867
EP 543991	A1	E		G06F-012/08	Based on patent WO 9222867
Designated States (Regional): BE CH DE ES FR GB IT LI NL SE					
JP 5509189	W			G06F-012/08	Based on patent WO 9222867
AU 658914	B			G06F-012/08	Previous Publ. patent AU 9222474 Based on patent WO 9222867
US 5442571	A		9	G06F-012/08	Cont of application US 91716397
EP 543991	B1	E		G06F-012/08	Based on patent WO 9222867
Designated States (Regional): BE CH DE ES FR GB IT LI NL SE					
DE 69229667	E			G06F-012/08	Based on patent EP 543991 Based on patent WO 9222867
TW 219986	A			G06F-013/00	
IL 102001	A			G06F-012/06	
CA 2088779	C			G06F-012/02	
KR 9605443	B1			G06F-012/08	

Computer performance improvements using simulated cache **associativity** ...

...**simulates effect of set associative** cache **by detecting** cache **misses and re-mapping pages on main memory**

...Abstract (Basic): The computer system is operated to simulate the effect of a set associative **cache** by detecting **cache** misses and remapping pages in the main memory. This enables memory references which would have caused thrashing to coexist in the **cache** .
...

...a different physical page frame but remain the same virtual address. This is accomplished by **updating** the page mapping tables to reflect the new physical location of the page, and copying the data from the **old** page frame to the **new** one

...Abstract (Equivalent): the method comprising the steps of accessing the **cache** to obtain a number of pages of data using the main memory addresses to identify a number of **cache** locations at which the pages may be stored, and detecting each of a number of **cache** misses comprising **cache** accesses resulting in the pages not being found at the identified locations. A preselected subset of the main memory addresses used in the **cache** misses is stored. The subset comprises more than one and less than the total number of the **cache** misses...

...USE - For operating **computer** system having processor, **main** memory for storing data, and **cache** for storing data corresp. to the data stored at selected main memory addresses...

...Title Terms: **CACHE** ;

International Patent Class (Main): **G06F-012/02** ...

... **G06F-012/06** ...

... **G06F-012/08** ...

... **G06F-013/00**

32/3,K/13 (Item 13 from file: 350)
DIALOG(R)File 350:Derwent WPIX

008441726 **Image available**
WPI Acc No: 1990-328726/199044
XRPX Acc No: N90-251678
Data processing system with queue mechanism - uses priority select unit, fast queue mechanism, main memory and cache **memory**
Patent Assignee: IBM CORP (IBMC); INT BUSINESS MACHINES CORP (IBMC)
Inventor: EMMA P G; KNIGHT J W; POMERENE J H; RECHTSCHAF R N; SPARACIO F J; RECHTSCHAFFEN R N
Number of Countries: 004 Number of Patents: 003
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 394620	A	19901031	EP 90101982	A	19900201	199044	B
US 5155831	A	19921013	US 89342493	A	19890424	199244	
EP 394620	A3	19920708	EP 90101982	A	19900201	199334	

Priority Applications (No Type Date): US 89342493 A 19890424
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 394620	A				
Designated States (Regional): DE FR GB					
US 5155831	A		12	G06F-009/06	

... **uses priority select unit, fast queue mechanism, main memory and** cache **memory**

...Abstract (Basic): any processor issues a store action to a line of memory in the queue, the **old** data is overwritten with the **new** data ...

...used, to be removed, when a processor accesses a line of data not in its **cache** (C1, C2...CN), a **cache** miss occurs and it is necessary to fetch the line from main memory. Such fetches...

...ADVANTAGE - Fast and efficient handling of live **data changes** . (14pp Dwg.No.2/5)
...Abstract (Equivalent): any processor issues a store action to a line of memory in the queue, the **old** data is overwritten with the **new** data. If the queue does not currently have a corresponding entry, a new entryu is...

...used, to be removed. An entry that is to be removed is first used to **update** the line corresponding to it in main memory. After the changes held in the entry...
...removed are applied to the old value of the line (from main memory) and the **updated** value is put back into main memory, the entry in the queue is removed by maring it ''empty''. When a processor accesses a line of data not in its **cache** , a **cache** loss occurs and it is necessary to fetch the line from main memory. Such fetches...

...another. USE/ADVANTAGE - Enables reduced traffic to main memory where multiprocessor system uses store-in **caches** .
(

...Title Terms: **CACHE** ;
International Patent Class (Main): **G06F-009/06**
International Patent Class (Additional): **G06F-009/312** ...

... **G06F-009/46** ...

... **G06F-012/02** ...

... **G06F-012/08**

32/3,K/14 (Item 14 from file: 350)
DIALOG(R)File 350:Derwent WPIX

008229116 **Image available**
WPI Acc No: 1990-116117/199015
XRPX Acc No: N90-089930

Pipelined single port buffer updated **method - allowing parallel operations by only** updating buffer **after beginning and end of contiguous set of pixel is determined**

Patent Assignee: SILICON GRAPHICS INC (SILI-N); SILICON GRAPHICS IN (SILI-N)
Inventor: HANNAH M R
Number of Countries: 014 Number of Patents: 007
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 9002990	A	19900322				199015	B
US 4951232	A	19900821	US 88243427	A	19880912	199036	
EP 433373	A	19910626	EP 89910565	A	19890911	199126	
JP 4506717	W	19921119	JP 89509882	A	19890911	199301	
			WO 89US3938	A	19890911		
EP 433373	B1	19960529	EP 89910565	A	19890911	199626	
			WO 89US3938	A	19890911		
DE 68926571	E	19960704	DE 626571	A	19890911	199632	
			EP 89910565	A	19890911		
			WO 89US3938	A	19890911		
KR 131820	B1	19980424	WO 89US3938	A	19890911	200001	
			KR 90700983	A	19900511		

Priority Applications (No Type Date): US 88243427 A 19880912
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 9002990	A		31		
Designated States (National): JP KR					
Designated States (Regional): AT BE CH DE FR GB IT LU NL SE					
US 4951232	A		11		
EP 433373	A				
Designated States (Regional): AT BE CH DE FR GB IT LI LU NL SE					
JP 4506717	W		7	G06F-015/72	Based on patent WO 9002990
EP 433373	B1	E	15	G06T-015/10	Based on patent WO 9002990
Designated States (Regional): CH DE FR LI					
DE 68926571	E			G06T-015/10	Based on patent EP 433373
					Based on patent WO 9002990
KR 131820	B1			G06F-001/04	

Pipelined single port buffer updated **method...**
...**allowing parallel operations by only** updating buffer **after beginning**

and end of contiguous set of pixel is determined

...Abstract (Basic): In the graphics **update** controller, graphics data is received from the main CPU via the bus (6) and processed...

...the ALU system (3) to produce new z-values to be stored in the z- **buffer** (11). The ALU compares, for each pixel, the **new** and **old** values of Z for the piexel. This comparison is performed serially across a row of ...

...The Z **buffer** is only **updated** once a 'fail' state occurs, hence the computation and Z **buffer** **writing** can occur in parallel with frame **buffer** **updating** .

...Abstract (Equivalent): In a **computer** **controlled** display system for presenting to an observer a representation of an image on a display apparatus, said system having a Z- **buffer** (11) for storing Z values and having a frame **buffer** (10) for storing pixel values, a method for **updating** said Z- **buffer** with **new** Z values to replace **old** Z-values comprising: receiving a pixel value and a new Z value for each pixel...

...an old Z value and an old pixel value being stored respectively in said Z- **buffer** (11) and said frame **buffer** (10), said Z-values representing a relative distance to said observer of a pixel displayed on said display apparatus; performing a Z comparison for each **new** Z value by comparing said **old** Z value with said **new** Z value for each pixel location, said Z comparison being performed sequentially in one direction...

...said old Z value; said method being characterised by the following steps: determining a first **update** pixel location, said first **update** pixel location being the location of where to begin an **updating** of said Z- **buffer** (11) with new Z values for a contiguous group of pixel locations having a second condition, said step of determining the first **update** pixel location including determining when two consecutive first and second Z comparisons for a first...

...produce a first and a second condition respectively, said second pixel location being said first **update** pixel location, determining a last **update** pixel location, said last **update** pixel location being the location of where to end said **updating** of said Z- **buffer** (11), said step of determining the last **update** pixel location including determining when two consecutive third and fourth Z comparisons for a third...

...produce a second and a first condition respectively, said third pixel location being said last **update** pixel location; **updating** said Z- **buffer** (11) with the new Z values for said contiguous group of pixel locations after determining said last **update** pixel location, said **updating** being performed by all Z values having pixel locations from and including said first **update** pixel location to and including said last **update** pixel location...

...Abstract (Equivalent): The Z- **buffer** is **updated** only after the Z comparison produces a combination of a fail condition for the current ...

...a contiguous (uninterrupted by pixels with a fail condition) plurality

of pixel locations will require **updating** , unless Z values are unusually erratic...

...The Z- **buffer** will be **updated** for the contiguous group of pixel locations having a pass condition, which contiguous group ends with the pixel location immediately preceding the current pixel location. Such **updating** of the Z- **buffer** occurs only after the combination of a fail condition which follows a pass condition. (11pp)
...Title Terms: **BUFFER ;**
International Patent Class (Main): **G06F-001/04** ...

... **G06F-015/72**
International Patent Class (Additional): **G06F-003/15** ...

... **G06F-003/153**

32/3,K/15 (Item 15 from file: 350)
DIALOG(R)File 350:Derwent WPIX

008228969 **Image available**
WPI Acc No: 1990-115970/199015
XRPX Acc No: N90-089828
Clearing region of Z- buffer **and screen -** writes **several bits into frame** buffer **to invalidate Z values for pixel locations in region selected for clearing**
Patent Assignee: SILICON GRAPHICS INC (SILI-N); SILICON GRAPHICS IN (SILI-N)
Inventor: HANNAH M R
Number of Countries: 014 Number of Patents: 006
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 9002780	A	19900322				199015	B
US 5038297	A	19910806	US 90614981	A	19901116	199134	
EP 551251	A1	19930721	EP 89910803	A	19890912	199329	
			WO 89US3924	A	19890912		
EP 551251	A4	19930811	EP 89910803	A	19890000	199527	
EP 551251	B1	19960207	EP 89910803	A	19890912	199610	
			WO 89US3924	A	19890912		
DE 68925651	E	19960321	DE 625651	A	19890912	199617	
			EP 89910803	A	19890912		
			WO 89US3924	A	19890912		

Priority Applications (No Type Date): US 88243789 A 19880913; US 90527644 A 19900522; US 90614981 A 19901116
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 9002780	A				
Designated States (National): JP KR					
Designated States (Regional): AT BE CH DE FR GB IT LU NL SE					
EP 551251	A1	E	2	C09G-001/02	Based on patent WO 9002780
Designated States (Regional): AT BE CH DE FR GB IT LI LU NL SE					
EP 551251	B1	E	13	G09G-001/16	Based on patent WO 9002780
Designated States (Regional): DE FR GB					
DE 68925651	E			G09G-001/16	Based on patent EP 551251
					Based on patent WO 9002780

Clearing region of Z- buffer **and screen...**

... writes **several bits into frame** buffer **to invalidate Z values for pixel locations in region selected for clearing**

...Abstract (Basic): The method clears a region of the Z- **buffer** in a raster scanned, **computer controlled** video display system having a frame **buffer** (15) and a Z- **buffer** (2). The frame **buffer** and the Z- **buffer** operate at two different speeds such that more bits of **data** may be **changed** during a given period of time in the frame **buffer** than in the Z- **buffer** .

...

...The method includes the step of **writing** a plurality of bits into pixel locations of the frame **buffer** , which pixel locations will be cleared in the Z- **buffer** and on the screen on the video display. . These bits which are **written** into the frame **buffer** will invalidate the Z values for the pixel locations in the region selected for clearing

...Abstract (Equivalent): In a raster scanned, **computer controlled** display system having a plurality of pixel locations, a frame **buffer** (15) and a Z- **buffer** (2), said frame **buffer** (15) and said Z- **buffer** (2) operating at two different speeds such that more bits of **data** may be **changed** during a given period of time in said frame **buffer** (15) than in said Z- **buffer** (2), a method for clearing at least a region of the Z- **buffer** (2), said method comprising the steps of: establishing in a portion of the frame **buffer** (15) a Z invalidity memory space (16), said Z invalidity memory space (16) including a storage bit location for each pixel location of the display, **writing** an invalidity bit into the storage bit locations corresponding to the region of the display...

...value for a current pixel location; checking the invalidity bit for said current pixel location; **updating** the pixel value and Z value of said current pixel location, said **updating** occurring without comparing the **old** Z value to the **new** Z value if said invalidity bit for said current pixel location is set; and disabling...

...Abstract (Equivalent): The method for clearing a region of the Z- **buffer** and hence a corresp. region of the screen of a video display appts. is disclosed for use in a raster scan, **computer controlled** video display system for presenting a representation a three-dimensional object on the video display appts., such as a computer monitor, to an observer of the video display appts.. The **computer controlled** video display system includes a Z- **buffer** for storing Z values and a frame **buffer** for storing pixel values...

...The frame **buffer** and the Z- **buffer** operates at two different speeds becuase more bits of **data** may be **changed** during a given period of time in one **buffer** than in the other **buffer** . Specifically, more bits of data and/or more memory storage locations (corresp. to more pixel locations) may be changed during a given period of time in the frame **buffer** than in the Z- **buffer** . (10pp)

...Title Terms: **BUFFER ;**

International Patent Class (Additional): **G06F-012/06** ...

... **G06F-015/72**

32/3,K/16 (Item 16 from file: 350)
DIALOG(R)File 350:Derwent WPIX

007970407 **Image available**
WPI Acc No: 1989-235519/198933
XRPX Acc No: N89-179419

Computer system comprising hierarchically organised memory - includes foreground memory having validity and modification bits for indicating valid information **and occurrence of** modification

Patent Assignee: DIGITAL EQUIP CORP (DIGI); PHILIPS GLOEILAMPENFAB NV (PHIG); NIPPON DIGITAL EQUIP KK (DIGI)
Inventor: FELDBRUGGE F H J; FELDBRUGGE F H
Number of Countries: 005 Number of Patents: 005
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 328172	A	19890816	EP 89200109	A	19890119	198933	B
NL 8800158	A	19890816				198936	
US 5241639	A	19930831	US 89300403	A	19890119	199336	
EP 328172	B1	19940302	EP 89200109	A	19890119	199409	
DE 68913316	E	19940407	DE 613316	A	19890119	199415	
			EP 89200109	A	19890119		

Priority Applications (No Type Date): NL 88158 A 19880125
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 328172	A	E	6		
Designated States (Regional): DE FR GB					
US 5241639	A		5	G06F-012/08	
EP 328172	B1	E	7	G06F-012/08	
Designated States (Regional): DE FR GB					
DE 68913316	E			G06F-012/08	Based on patent EP 328172

... **includes foreground memory having validity and modification bits for indicating valid** information **and occurrence of** modification

...Abstract (Basic): the foreground memory, there is provided a validity bit (valid) in order to indicate valid **information** storage and a **modification** bit (dirty) in order to indicate a modification effected ...

...is provided a registration memory which comprises a row of further memory locations in which **write** operations can be performed in a predetermined address sequence and which can be directly read...

...ADVANTAGE - Short period of time required for copying **modified information** into main memory.
...Abstract (Equivalent): therein, wherein there is provided a registration memory (62) which comprises memory locations in which **write** operations are performed in a predetermined first address sequence in order to store in each...
...Abstract (Equivalent): In response to a **write** transaction, data is **written** to a location in a foreground memory that is part of the **cache** . A modified bit is set in the foreground memory indicating that the data must be **written** back to main memory. A registration bit is checked in the foreground memory location to determine whether the address is registered. The location of the foreground memory is

written to a registration memory in the **cache** if the registration bit is not set. The registration bit is set in the foreground memory, in conjunction with **writing** the address of the memory location to the registration memory, to indicate that the address...

...to a subsequent transaction, the foreground memory location can be selected to store new required **data** . The **modified** bit is checked to determine whether the data stored in the foreground memory location must be **written** back to main memory before the location can be used to store the new data. The data stored in the foreground memory location is **written** back to the main memory if the modified bit is set. The foreground memory location is filled with the new data while the **cache** address of the **new** data and the **previous** state of the registration but are retained in the registration memory. The modified bit is...

...USE/ADVANTAGE - Method for operating a **cache** memory for **temporary storage** of information in computer systems. Registration memory can be directly addressed, doing away with complex...

International Patent Class (Main): **G06F-012/08**

32/3,K/17 (Item 17 from file: 350)

DIALOG(R)File 350:Derwent WPIX

007464827 **Image available**

WPI Acc No: 1988-098761/198814

XRPX Acc No: N88-074825

Digital computer **memory appts. for eviction** control **system - has translator with control field including flipper bit, and eviction pending bit for each address field**

Patent Assignee: AMDAHL CORP (AMDA)

Inventor: HANSON D L; WOFFINDEN G A

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 4731739	A	19880315	US 85789035	A	19851017	198814	B

Priority Applications (No Type Date): US 83527676 A 19830829; US 85789035 A 19851017

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 4731739	A		21		

Digital computer **memory appts. for eviction** control **system...**

...Abstract (Basic): translator for translating logical addresses to system addresses. The translator includes a translation look aside **buffer** having a system address store with two fields storing system addresses corresp. to each translated...

...When the TLB is **updated** with a **new** translation, the **previous** active address field is marked with an eviction pending bit and the previous inactive address field is **written** over with the new translation and marked active. After eviction from the intermediate store of...

International Patent Class (Additional): **G06F-012/12**

32/3,K/18 (Item 18 from file: 347)
DIALOG(R)File 347:JAPIO

05888560 **Image available**
DATA COMMUNICATION DEVICE

PUB. NO.: 10-171660 [JP 10171660 A]
PUBLISHED: June 26, 1998 (19980626)
INVENTOR(s): SATO HIROYUKI
IKEDA SABURO
FUJII TAKEHIRO
MORI MARIKO
MIZOBE MASATOSHI
ATSUTA HIROMI
YOSHIDA CHIZUKO
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 08-332575 [JP 96332575]
FILED: December 13, 1996 (19961213)

INTL CLASS: **G06F-009/445** ; H04L-029/08

ABSTRACT

...and controlling the communication of data via a data communication means in preference to the **transfer** of data via a data **transfer** means...

...SOLUTION: The communication control processing is executed in preference to the data **transfer** processing. That is, a CPU 6 copies an old program to a primary saving area...

... and edits a communication control program while carrying on the communication control processing. Then an **updated** program is **transferred** to the communication control program storage area of an EPROM 10 and **updated** there. At the same time, the **changed** differential **data** stored in a **RAM** 8 are **transferred** to the **changed** differential **information** storage areas of an EPROM 9 and then **transferred** to other data communication devices 1a and 1b from a communication I/O part 5. The devices 1a and 1b **update** the **old** programs to the **new** ones and **transfer** the **changed** differential **data** to a subordinate data communication device 1c.

32/3,K/19 (Item 19 from file: 347)
DIALOG(R)File 347:JAPIO

05468959 **Image available**
IMAGE FORMING DEVICE

PUB. NO.: 09-083759 [JP 9083759 A]
PUBLISHED: March 28, 1997 (19970328)
INVENTOR(s): OTANI MASAYUKI
APPLICANT(s): RICOH CO LTD [000674] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 07-240137 [JP 95240137]
FILED: September 19, 1995 (19950919)

INTL CLASS: H04N-001/21; B41J-005/30; B41J-029/38; G03G-021/00; **G06F-003/12**

...JAPIO KEYWORD:Charge **Transfer** Elements, CCD & BBD); R116 (ELECTRONIC MATERIALS...

ABSTRACT

PROBLEM TO BE SOLVED: To enable **updating** of data and to extend the limit of the number of times of rewriting by...

...a means copying an operating condition set value in another block at the time of **updating** the operating condition and deleting an original block after the completion of copying...

...SOLUTION: Whether the copying from an **old** small block to a **new** small block is completed or not is judged. If the copying is completed, a copy resumption processing is performed (S1). Namely, when the copying is completed, the **data** **changed** by a key operation is stored in a **RAM** for work (S2). When the shift of a screen is generated (S3), the presence or absence of the **change** of **data** is judged (S4 to S8). When even one **data** is **changed** , a **COPY** flag to be a new flag is set to O (S9, 10). Subsequently, the data of the present small block (**old**) is copied in the next small block (**new**) (S11). After the completion of the copying, the CPEND of the status flag of the...

32/3,K/20 (Item 20 from file: 347)

DIALOG(R)File 347:JAPIO

05307579 **Image available**

COMMUNICATION KARAOKE SYSTEM

PUB. NO.: 08-263079 [JP 8263079 A]
PUBLISHED: October 11, 1996 (19961011)
INVENTOR(s): ITO SHIGEYOSHI
TSURUMI KANEHISA
MURAI YUICHI
APPLICANT(s): YAMAHA CORP [000407] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 07-069925 [JP 9569925]
FILED: March 28, 1995 (19950328)

INTL CLASS: G10K-015/04; **G06F-009/445** ; H04M-011/08; **G06F-013/00**

ABSTRACT

... or a version-up is executed, a system program is distributed together with its version **management** information to each KARAOKE **terminal** from a host computer. A CPU 22 in the KARAOKE terminal stores the received system program as well as the version management information in a hard **disk** 24. When an **old** version system program is already stored, a **new** system program is added. Moreover, the version management information is **written** into a version management table in the **disk** 24. Starting with a next time, the latest program is executed while referring to the...

32/3,K/21 (Item 21 from file: 347)

DIALOG(R)File 347:JAPIO

04772238 **Image available**
INFORMATION PROCESSOR AND CONTROL METHOD

PUB. NO.: 07-064838 [JP 7064838 A]
PUBLISHED: March 10, 1995 (19950310)
INVENTOR(s): NAGASAKI HIDENORI
APPLICANT(s): TOSHIBA CORP [000307] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 05-215665 [JP 93215665]
FILED: August 31, 1993 (19930831)

INTL CLASS: **G06F-012/00 ; G06F-012/00**

ABSTRACT

PURPOSE: To easily recognize **new / old** change contents of each item at the time of display of individual information and to...

...CONSTITUTION: At each time of the **change** of individual **information** , new information is stored as the **updated** information in an individual information storage area 13a of a **RAM** 13, and old information is stored there in the order of **change** as the history **information** . A CPU 11 displays individual information on a display part 16 in such form based on the **updated** information and history information stored in the individual information storage area 13a that **new / old** change contents of each item can be understood. While the period when the history information...

32/3,K/22 (Item 22 from file: 347)
DIALOG(R)File 347:JAPIO

04384360 **Image available**
STORAGE DEVICE

PUB. NO.: 06-028260 [JP 6028260 A]
PUBLISHED: February 04, 1994 (19940204)
INVENTOR(s): KASHIMA RIKA
NAKAMURA SHUNICHIRO
MINEMURA HARUMI
SHIONO KATSUMI
HAYAKAWA TAKAYUKI
YOSHIMORI MIKIO
HIGUCHI MASAHIRO
APPLICANT(s): MITSUBISHI ELECTRIC CORP [000601] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 04-195004 [JP 92195004]
FILED: July 22, 1992 (19920722)
JOURNAL: Section: P, Section No. 1735, Vol. 18, No. 247, Pg. 47, May 11, 1994 (19940511)

INTL CLASS: **G06F-012/08 ; G06F-003/06**

ABSTRACT

PURPOSE: To speed up the speed of a data **writing** process by decreasing the frequency of access to a **disk** device at the time of data **writing** to a **disk** array device...

...CONSTITUTION: When data are **written** in the **disk** array device 1, the

data in a **disk** **cache** 13 provided on the main storage device 12 of a **main** **computer** body 10 are **updated** ; and old data and old check data are stored in an old data/old check data **cache** 15 at this time and then the exclusive OR arithmetic unit 3 of the **disk** array device 1 generates new check data on the basis of **new** data and the **old** data and **old** check data in the old data/ **old** check data **cache** 15, and stores the **new** data in a **disk** device 2a of the **disk** array device 1 and the new check data in a **disk** device 2d.

32/3,K/23 (Item 23 from file: 347)
DIALOG(R)File 347:JAPIO

04205770 **Image available**
KEYBOARD DEVICE

PUB. NO.: 05-197470 [JP 5197470 A]
PUBLISHED: August 06, 1993 (19930806)
INVENTOR(s): YAMAMOTO KATSUHISA
APPLICANT(s): RICOH CO LTD [000674] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 04-032548 [JP 9232548]
FILED: January 22, 1992 (19920122)
JOURNAL: Section: P, Section No. 1645, Vol. 17, No. 621, Pg. 149, November 16, 1993 (19931116)

INTL CLASS: **G06F-003/023 ; G06F-003/03**

ABSTRACT

PURPOSE: To provide a keyboard device which respective users can easily use by storing **new** data without **updating** **original** layout data at the time of **updating** an item key layout...

... auxiliary function key 10 so as to designate change contents from an operation panel 8. **Changed** **data** is stored in **RAM** c with a page number. At this time, original item key layout data is left in **RAM** b or a memory card m as it is without being **updated** . When the user wants the layout returned to the original one, he can return it...

... layout without down-loading from a host device 1 again. When auxiliary information, four-bit **data** which indicates that **change** is not desired, is added at the side of the host device 1, an item...

32/3,K/24 (Item 24 from file: 347)
DIALOG(R)File 347:JAPIO

03812420 **Image available**
DISK DRIVER DEVICE

PUB. NO.: 04-177520 [JP 4177520 A]
PUBLISHED: June 24, 1992 (19920624)
INVENTOR(s): WATANABE KAZUNORI
APPLICANT(s): NEC IC MICROCOMPUT SYST LTD [470861] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 02-305233 [JP 90305233]

FILED: November 09, 1990 (19901109)
JOURNAL: Section: P, Section No. 1435, Vol. 16, No. 493, Pg. 84, October 13, 1992 (19921013)

DISK DRIVER DEVICE

INTL CLASS: **G06F-003/06** ; **G06F-012/00** ; G11B-020/00

ABSTRACT

PURPOSE: To minimize the damage even if an accident occurs in the course of **writing** **data** by **changing** its **file** name in the case there is **old** data of the same file name, **writing** **new** data as a **new** file, and erasing the **old** data after **writing** is finished...

...CONSTITUTION: A **disk** driver 12 is mounted together with a CPU 11 in a **computer** **main** body 9. In such a state, a **write** instruction from the CPU 11 is received by this **disk** driver 12, a processing instruction to a **disk** controller 13 is generated, and data is **written** in a **disk** medium 14. In such a case, since the **disk** driver 12 is set in the **computer** **main** body 9, an existing **disk** device 10 can be utilized as it is. In such a way, an operator can...

... case an accident such as a power failure, etc., is generated in the course of **writing** the data, since old data remains behind, the work time required for release of the...

32/3,K/25 **(Item 25 from file: 347)**
DIALOG(R)File 347:JAPIO

03114875 **Image available**
ARITHMETIC PROCESSING SYSTEM FOR PICTURE DATA

PUB. NO.: 02-090375 [JP 2090375 A]
PUBLISHED: March 29, 1990 (19900329)
INVENTOR(s): NOGUCHI YASUSHI
YASUDA KAZUNORI
SUGIURA MARI
APPLICANT(s): SONY CORP [000218] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 63-243343 [JP 88243343]
FILED: September 28, 1988 (19880928)
JOURNAL: Section: P, Section No. 1066, Vol. 14, No. 293, Pg. 152, June 25, 1990 (19900625)

INTL CLASS: **G06F-015/66** ; G09G-005/36; H04N-001/387; H04N-007/08

ABSTRACT

PURPOSE: To shorten the arithmetic processing time by reading only the **new** **original** picture element data out of a video memory among plural original picture element data used for calculation of a certain process picture element data and **writing** the read-out picture element data into a high-speed arithmetic access memory...

...CONSTITUTION: The video signals received from an input terminal 9 are **written** into a video memory 6 via an A/D converter 10. Then the video signals...

...monitor picture receiver 8 via a D/A converter 7. The still picture data is **written** into the memory 6 and magnified or reduced under the **control** of a **computer** 1 via a **buffer** memory 14 and a digital signal processing circuit 15. Then the magnified or reduced picture...

... via a transmission line 13 after a communication process. The process circuit 12 reads the **new** **original** picture element data out of the memory 6 except those original picture element data used...

... preceding the previous one. The picture element data read out of the memory 6 is **written** into the memory 14 and undergoes an arithmetic process.

32/3,K/26 (Item 26 from file: 347)
DIALOG(R)File 347:JAPIO

03002719 **Image available**
SYSTEM EXPANDING SYSTEM

PUB. NO.: 01-300319 [JP 1300319 A]
PUBLISHED: December 04, 1989 (19891204)
INVENTOR(s): SEKI MAKI
TAKEGAHARA TAKASHI
MORIZAKI KAZUHIKO
APPLICANT(s): FANUC LTD [419041] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 63-132145 [JP 88132145]
FILED: May 30, 1988 (19880530)
JOURNAL: Section: P, Section No. 1009, Vol. 14, No. 92, Pg. 117, February 20, 1990 (19900220)

INTL CLASS: **G06F-001/00 ; G06F-003/00**

ABSTRACT

PURPOSE: To realize the addition of a **new** option even though an **old** option is already connected by securing such a constitution where a data **transfer** control part of the new option **transfers** the data given from a **main** **computer** device to the old option via a 2nd connector in case an address space of...

...to a connector 12c of an old option 12 via a bus. Then a data **transfer** control part 13a makes the automatic programming device 11 have an access to a built-in shared **RAM** in case an address space of a new option is designated by the device 11. While the part 13a **transfers** the data received from the device 11 to the option 12 via the connector 13d...

32/3,K/27 (Item 27 from file: 347)
DIALOG(R)File 347:JAPIO

02697155 **Image available**
SCREEN EDITING SYSTEM

PUB. NO.: 63-314055 [JP 63314055 A]

PUBLISHED: December 22, 1988 (19881222)
INVENTOR(s): HIGUCHI KAZUHIRO
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 62-150472 [JP 87150472]
FILED: June 17, 1987 (19870617)
JOURNAL: Section: E, Section No. 744, Vol. 13, No. 160, Pg. 53, April 18, 1989 (19890418)

INTL CLASS: H04M-003/22; **G06F-009/00** ; H04M-003/00

ABSTRACT

PURPOSE: To reduce the input error by displaying maintenance terminal display **data** before **change** or correction and that after change or correction with characters in contradistinction to each other...

...CONSTITUTION: A key corresponding to maintenance terminal display **data** to be **changed** or corrected is inputted from a maintenance terminal 1, and a search means 6 searches...

... address table 5 to obtain storage position information of maintenance terminal display data. A read/ **write** means 7 **writes** storage position information in an address part 8a of a data editing **buffer** 8 and **writes** maintenance **terminal** display data of a **main** memory 3 in a **new** image part 8b and an **old** image part 8c of the data editing **buffer** . A **write** means 9 rewrites contents of the new image part 8b in accordance with the operation on the maintenance **terminal** 1, and a display **control** means 10 displays contents of the **new** image part 8b and the **old** image part of the data editing **buffer** 8 with characters on the maintenance terminal 1. Thus, **data** is easily **changed** or corrected without error.

32/3,K/28 (Item 28 from file: 347)
DIALOG(R)File 347:JAPIO

02509132 **Image available**
PROGRAM **UPDATING** CONTROL SYSTEM

PUB. NO.: 63-126032 [JP 63126032 A]
PUBLISHED: May 30, 1988 (19880530)
INVENTOR(s): SHIMIZU TATEO
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 61-271239 [JP 86271239]
FILED: November 14, 1986 (19861114)
JOURNAL: Section: P, Section No. 769, Vol. 12, No. 380, Pg. 97, October 12, 1988 (19881012)

PROGRAM **UPDATING** CONTROL SYSTEM

INTL CLASS: **G06F-009/06** ; **G06F-011/28** ; **G06F-013/00**

ABSTRACT

... the efficiency of transmission between a host computer and a terminal equipment by replacing an **old** program by a **new** program only by the transmission of a module component...
... a program which is transmitted from the host computer 5 is temporarily

stored in a **buffer** memory 4 through a **terminal control** device 7. In order to replace the old module in each program using a module...

...by a new module on the basis of a module control data 9 in an **updating** module control file 8 connected to the device 7, a program 2 in a program ...

...the module 3 in the program are replaced by a new module stored in the **buffer** memory 4. Consequently, replacement to the new program can be attained only by the transmission...

32/3,K/29 (Item 29 from file: 347)
DIALOG(R)File 347:JAPIO

02409148 **Image available**
METHOD OF SETTING/READING TERMINAL IDENTIFICATION CODE

PUB. NO.: 63-026048 [JP 63026048 A]
PUBLISHED: February 03, 1988 (19880203)
INVENTOR(s): AIDA KENJU
OKAZAKI TETSUO
APPLICANT(s): NIPPON TELEGR & TELEPH CORP <NTT> [000422] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 61-167853 [JP 86167853]
FILED: July 18, 1986 (19860718)
JOURNAL: Section: E, Section No. 629, Vol. 12, No. 234, Pg. 16, July 05, 1988 (19880705)

INTL CLASS: H04L-011/00; **G06F-013/00**

ABSTRACT

... ID when the terminal is changed by making setting of identification codes (ID) of plural **terminals** on a table **controlled** by a controlling section instead of setting directly to special hardware possessed by terminals...
...which (ID) is set is transmitted to the terminal section 1. The terminal section 1 **writes** the above-mentioned ID in an **RAM** on the terminal and makes the terminal usable state. A the time of **updating** ID by operation of the terminal section 1, the terminal transmits a telegraphic message of request for ID **updating** in which **new** and **old** ID are set to the controlling section 1. On receiving the telegraph, the controlling section 2 regards it an error when the **old** and **new** ID do not conform and when the flag indicating in use is off. When all checks OK, ID of the table is **updated** , and the new ID is transmitted to the terminal section 1.

32/3,K/30 (Item 30 from file: 347)
DIALOG(R)File 347:JAPIO

01966267 **Image available**
PICTURE RETRIEVING DEVICE

PUB. NO.: 61-180367 [JP 61180367 A]
PUBLISHED: August 13, 1986 (19860813)
INVENTOR(s): TSUKAMOTO MASAYOSHI

APPLICANT(s): TOSHIBA CORP [000307] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 60-020713 [JP 8520713]
FILED: February 04, 1985 (19850204)
JOURNAL: Section: P, Section No. 532, Vol. 11, No. 2, Pg. 106, January 06, 1987 (19870106)

INTL CLASS: **G06F-015/30** ; **G06F-015/40** ; **G06F-015/62** ; G11B-027/10; H04N-001/00

ABSTRACT

... To attain the trace-back retrieval of picture information from the latest one despite the **change** of these **information** , by reversing both registering and displaying orders between **new** and **old** pieces of picture information corresponding to the same retrieval information...
... in the order of larger address numbers of a retrieval table set on a magnetic **disk** memory 11. Then these account numbers are **written** to an optical **disk** memory 10 in the order of earlier coincidence with the retrieval numbers. The number of...

... the account numbers are calculated when all retrieving jobs are over. These calculated numbers are **written** on a retrieval **buffer** in the form of the number of registered seal stamps. Then the addresses on the...

32/3,K/31 **(Item 31 from file: 347)**
DIALOG(R)File 347:JAPIO

01551360 **Image available**
UPDATING METHOD OF DATA

PUB. NO.: 60-029860 [JP 60029860 A]
PUBLISHED: February 15, 1985 (19850215)
INVENTOR(s): TAKEZOE FUMIHIKO
APPLICANT(s): FUJI ELECTRIC CO LTD [000523] (A Japanese Company or Corporation), JP (Japan)
FUJI FACOM CORP [470926] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 58-131190 [JP 83131190]
FILED: July 18, 1983 (19830718)
JOURNAL: Section: P, Section No. 368, Vol. 09, No. 156, Pg. 26, June 29, 1985 (19850629)

UPDATING METHOD OF DATA

INTL CLASS: **G06F-013/16** ; **G06F-012/02** ; **G06F-015/16**

ABSTRACT

... the bit position of stored data in the address signal of a memory plane of **RAMs** , **updating** the contents of a bit, and designates the contents of the **update** upon when necessary...

...CONSTITUTION: A memory control circuit 10 when receiving a **data modification** command MDF from a bus 2 switches a switch (SW) 43 to connect a bus...

...enable state RE. Consequently, old data D(sub 0) is latched 31 in a data

23/3,K/1 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

009273816 **Image available**
WPI Acc No: 1992-401227/199249
XRPX Acc No: N92-305966
Reading and writing **data from and to tape storage medium - by transferring data between** buffer **memory and storage medium in successive** transfer **operations involving setting-up and** transfer **phases**
Patent Assignee: HEWLETT-PACKARD LTD (HEWP); HEWLETT-PACKARD CO (HEWP)
Inventor: RUSHTON N
Number of Countries: 004 Number of Patents: 004
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
GB 2256295	A	19921202	GB 929887	A	19920507	199249	B
DE 4215148	A	19921203	DE 4215148	A	19920508	199250	
US 5412780	A	19950502	US 92873426	A	19920427	199523	
GB 2256295	B	19950614	GB 929887	A	19920507	199527	

Priority Applications (No Type Date): GB 9111524 A 19910529
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
GB 2256295	A		35	G06F-005/06	
DE 4215148	A		19	G11B-005/00	
US 5412780	A		17	G06F-013/00	
GB 2256295	B		2	G06F-005/06	

Reading and writing **data from and to tape storage medium...**

...**by transferring data between** buffer **memory and storage medium in successive** transfer **operations involving setting-up and** transfer **phases**

...Abstract (Basic): involves the steps of transferring data to/from the storage appts., storing data in the **buffer** memory and transferring data between the **buffer** memory and storage medium in successive **transfer** operations each of which comprises a preparatory phase setting-up conditions for data **transfer** , and a **transfer** phase in which data is **transferred** continuously. The steps are ordered for any given item of data and the direction of data **transfer** depends on whether data is being **written** or read...

...Each **transfer** operation is instigated upon the amount of **buffer** memory ready to partake in the **transfer** process falling below a threshold value. The externally useful memory amount, for data **writing** , is the amount of **buffer** memory free to receive data and for data reading, is the amount of **buffer** memory with data to be **transferred** from the appts.. The threshold value is adaptively determined depending on utilisation of the **buffer** memory during a previously initiated **transfer** operation...

...ADVANTAGE - Improves **buffer** utilisation. Minimises waiting time to maximise data **transfer** rate...
...Abstract (Equivalent): or writing data to/from a storage medium using data storage apparatus that includes a **buffer** memory, and method comprising the steps of: a) transferring data to/from the said storage

apparatus, b) storing data in said **buffer** memory, c) transferring data between said **buffer** memory and said storage medium in successive transfer operations each of which comprises a preparatory...

...a threshold value, said externally-useful memory amount being, for data writing, the amount of **buffer** memory free to receive data, and for data reading, the amount of **buffer** memory with data to be transferred from the apparatus, wherein said threshold value governing the next instigation of a step (c) transfer operation is adaptively determined by measuring said **remaining** externally-useful **memory** amount during a previously-initiated step (c) transfer operation, and deriving said threshold value in dependence upon this measured amount to minimise the **remaining** said externally-useful **memory** amount present when, following instigation of a step (c) transfer operation, this step operation is...

...Abstract (Equivalent): medium uses a device data storage for reading and/or writing data that includes a **buffer** memory for smoothing the flow of data to/from the storage medium. Activation of the...

...storage medium is generally controlled in dependence upon the amount of data held in the **buffer** .

...

...In order to optimise utilisation of the **buffer** , it is proposed to adaptively set the threshold level at which such activation is effective. To this end, the utilisation of the **buffer** during a **previous** activation is monitored and a **new** threshold value is set in dependence on the monitored utilisation parameter

...Title Terms: **WRITING** ;

International Patent Class (Main): **G06F-005/06** ...

International Patent Class (Additional): **G06F-013/00**

?

Software file

```
ds; show files

Set      Items   Description
S1        2898   (CACHE? OR   TEMPORARY()(STORAGE OR  MEMORY? )
             OR BUFFER? OR RAM?)
S2       11861   (WRIT??? OR WRIT???()DISK? OR TRANSFER??? OR RECORD??? OR -
             ENCOD??? OR UPDAT???)
S3         484   (OLD OR ORIGINAL? OR PREVIOUS OR EARL??? OR OUTDAT???)(7N)-
             (NEW??? OR RECENT OR MODERN OR FRESH)
S4         762   (MODIF???  OR MODIFICAT??? OR AMEND??? OR AMENDM??? OR CHA-
             NG??? OR ADJUST??? OR ADJUSTM???)(3N)(COP??? OR DATA OR FILE??
              OR INFORMATION?? OR INFO? OR VERSION? )
S5       25645   (NODE? OR TERMINAL? OR COMPUTER? OR CLIENT? OR SERVER? OR -
             WORKSTATION?? OR STATION??)
S6          37   (PERSIST??? OR PERSEVER??? OR ENDUR??? OR LINGER??? OR REM-
             AIN??? OR STEAD??? OR PROLONG??? OR UNCHANG???)(3N)(STOR??? OR
              STORAGE? OR MEMOR??? OR HARD()DRIVE OR DISK? OR CD()ROM?? OR
             CDROM OR ROM?? OR FLASH? OR EPROM?? OR PROM??)
S7        4470   (MASTER? OR CONTROL??? OR MAIN OR PRIMARY? OR SUPERVIS??? -
             OR ADMINISTRA??? OR MANAG?)(5N) S5
S8           0   AU=(CHANDRASEKARAN, S?  OR CHANDRASEKARAN S?)
S9           0   AU=(BAMFORD, R? OR BAMFORD R? )
S10          0   AU=(BRIDGE, W? OR BRIDGE W?)
S11          0   AU=(BROWER, D? OR  BROWER D?)
S12          0   AU=(MACNAUGHTON, N?  OR MACNAUGHTON N?)
S13          1   AU=(CHAN, W? OR  CHAN W?)
S14          0   AU=(SRIHARI, V? OR SRIHARI V?)
S15          0   S13 AND S1
S16          0   S1(S)S2(S)S3(S)S4(S)S4(S)S6(S)S7
S17          0   S1 AND S2 AND S3 AND S4 AND S4 AND S6 AND S7
S18          7   S1(S)S2(S)S3
S19          0   S1(S)S2(S)S3(S)S4
S20          2   S6(S)S7
S21          5   S1(S)S2(S)S3(S)(S6 OR S7 OR S5)
S22          9   S18 OR S20 OR S21
S23          9   S22 NOT PY>1999
S24          3   S23 NOT PD=19981124:20041230
File 256:TecInfoSource 82-2004/Dec
         (c) 2004 Info.Sources Inc
?
```

24/3,K/1
DIALOG(R)File 256:TecInfoSource

02351245 DOCUMENT TYPE: Company

IBM Corp (351245)
1133 Westchester Ave
White Plains, NY 10604 United States
TELEPHONE: (914) 499-1900
TOLL FREE TELEPHONE NUMBER: (800) 426-4968
FAX: (800) 445-2426
HOMEPAGE: http://www.ibm.com
TICKER: NYSE : IBM

RECORD TYPE: Directory

CONTACT: Sales Department

ORGANIZATION TYPE: Corporation
EQUITY TYPE: Public
STATUS: Active

NUMBER OF EMPLOYEES: 315,889
SALES: 2,147,483,647
DATE FOUNDED: 1911
PERSONNEL: Palmisano, Samuel J, Chief Executive Officer; Palmisano, Samuel J, Chairperson; Donofrio, Nicholas M, VP Technology; Horn, Paul M, VP Research & Development; Joyce, John R, Chief Financial Officer; Kohnstamm, Abby F, VP Marketing; Elix, Doug T, VP; Harreld, J Bruce, VP ; Lawrie, Michael J, VP
REVISION DATE: 20040530

IBM Corporation was incorporated in the state of **New** York in 1911. **Originally** named Computing-Tabulating- **Recording** Company, the company adopted the name IBM (International Business Machines) in 1924. Its common stock...

...range of information technology products and services. IBM's pioneering products include the lock autograph **recorder** (1920), the rotary press (1924), the first vacuum tube **computer** (1952), random access **disk** storage (1957), the Fortran programming language (1957), the mainframe, the floppy **disk** (1971), the supermarket checkout **station** (1973), the ATM (1973), and the personal **computer** (1981). In more recent years, IBM reorganized to focus on customers and integrated business solutions...

24/3,K/2
DIALOG(R)File 256:TecInfoSource

01403571 DOCUMENT TYPE: Product

PRODUCT NAME: BIBLE-KJV for Windows (403571)

Linguist's Software Inc (418528)
PO Box 580
Edmonds, WA 98020-0580 United States
TELEPHONE: (425) 775-1130

RECORD TYPE: Directory

CONTACT: Sales Department

REVISION DATE: 20040216

...Software includes the complete text of the King James Version of the Holy Bible on **disks** . It includes both the **Old** and **New** Testaments plus the Apocrypha. The text is formatted for Windows (R) **Write** (R). The KJV is also offered as part of LS's scholars' packages.

24/3,K/3
DIALOG(R)File 256:TecInfoSource

01081825 DOCUMENT TYPE: Product

PRODUCT NAME: HDD Sheriff (081825)

JungSoft USA Inc (718149)
274-5 Seohyun-Dong Bundang-Gu, Seongnam-City
Kyunggi-Do, KR Korea
TELEPHONE: ( ) 823-17885124

RECORD TYPE: Directory

CONTACT: Sales Department

REVISION DATE: 20020430

...streamlines software re-installations and troubleshooting procedures. Tapping HDD Sheriff, system administrators can prevent hard **disk** damage and data corruption. The system restores data after a standard reboot. Administrators and technology instructors also can use HDD Sheriff to protect multi-user **computer** systems from viruses. Featuring an integrated BIOS, HDD Sheriff does not require an operating system...
...relocated files automatically at rebooting. The application saves files to a number of protected hard **disk** areas. Protection Area (C:) ignores **new** edits, saving **original** data. The data can be restored on rebooting. Data Area (D: or E:) is an unprotected hard **disk** zone, where system users can **update** and save files. The Backup Area is a storage area for deleted or edited data...
?

Techniques for Reducing Consistency-Related Communication in Distributed Shared-Memory Systems

JOHN B. CARTER
University of Utah
and
JOHN K. BENNETT and WILLY ZWAENEPOEL
Rice University

Distributed shared memory (DSM) is an abstraction of shared memory on a distributed-memory machine. Hardware DSM systems support this abstraction at the architecture level; software DSM systems support the abstraction within the runtime system. One of the key problems in building an efficient software DSM system is to reduce the amount of communication needed to keep the distributed memories consistent. In this article we present four techniques for doing so: software release consistency; multiple consistency protocols; write-shared protocols; and an update-with-timeout mechanism. These techniques have been implemented in the Munin DSM system. We compare the performance of seven Munin application programs: first to their performance when implemented using message passing, and then to their performance when running on a conventional software DSM system that does not embody the preceding techniques. On a 16-processor cluster of workstations, Munin's performance is within 5% of message passing for four out of the seven applications. For the other three, performance is within 29 to 33%. Detailed analysis of two of these three applications indicates that the addition of a function-shipping capability would bring their performance to within 7% of the message-passing performance. Compared to a conventional DSM system, Munin achieves performance improvements ranging from a few to several hundred percent, depending on the application.

Categories and Subject Descriptors: B.3.2 [**Memory Structures**]: Design Styles—*cache memories*; *shared memory*; virtual memory; C.1.2 [**Processor Architectures**]: Multiple Data Stream Architectures (Multiprocessors)—*interconnection architectures*; *parallel processors*; D.4.2 [**Operating Systems**]: Storage Management—*distributed memories*; *virtual memory*; D.4.4 [**Operating Systems**]: Communications Management—*network communication*; D.4.7 [**Operating Systems**]: Organization and Design—*distributed systems*; D.4.8 [**Operating Systems**]: Performance —*measurements*

This research was supported in part by the National Science Foundation under grants CDA-8619893, CCR-9010351, CCR-9116343, by the IBM Corporation under Research Agreement no. 20170041, by the Texas Advanced Technology Program under grants 003604014 and 003604012, and by a NASA Graduate Fellowship.
Authors' addresses: J. B. Carter, Department of Computer Science, University of Utah, 3190 Merrill Engineering Building, Salt Lake City, UT 84112; email: retrac@cs.utah.edu; J. K. Bennett and W. Zwaenepoel, Computer Systems Laboratory, Rice University, Houston, TX 77251-1892.

General Terms. Algorithms, Design, Performance

Additional Key Words and Phrases: Cache consistency protocols, distributed shared memory, memory models, release consistency, virtual shared memory

1. INTRODUCTION

1.1 Background

There are two fundamental models for parallel programming and for building parallel machines: shared memory and distributed memory or message passing. The *shared-memory model* is a direct extension of the conventional uniprocessor model wherein each processor is provided with the abstraction that there is but a single memory in the machine. An update to shared data, therefore, becomes visible to all the processors in the system. In contrast, in the *distributed-memory model* there is no single shared memory. Instead, each processor has a private memory to which no other processor has direct access. The only way for processors to communicate is through explicit *message passing*.

Distributed-memory machines are easier to build, especially for large configurations, because unlike shared-memory machines they do not require complex and expensive hardware cache controllers [Archibald and Baer 1986]. The shared-memory programming model is, however, more attractive since most application programmers find it difficult to program machines using a message-passing paradigm that requires them to partition data and manage communication explicitly. Using a programming model that supports a global address space, an applications programmer can focus on algorithmic development rather than on managing partitioned data sets and communicating values.

A distributed shared-memory (DSM) system provides a shared-memory programming model on a distributed-memory machine. Hardware DSM systems, e.g., DASH [Lenoski et al. 1990], support this abstraction at the architecture level; software DSM systems, such as Ivy [Li and Hudak 1989] and Munin [Carter et al. 1991], support this abstraction within the runtime system. Software DSM systems consist of the same hardware as that found in a distributed-memory machine, with the addition of a software layer that provides the *abstraction* of a single shared memory. In practice, each memory remains physically independent, and all communication takes place through explicit message passing performed by the DSM software layer. DSM systems combine the best features of shared-memory and distributed-memory machines. They support the convenient shared-memory programming model on distributed-memory hardware, which is more scalable and less expensive to build. However, although many DSM systems have been proposed and implemented (see Bal et al. [1992], Bershad et al. [1993], Chase et al. [1989], Dasgupta et al. [1990], Fleisch and Popek [1989], Li and Hudak [1989], and Minnich and Farber [1989]), achieving good performance on DSM systems for a sizable class of applications has proven to be a major challenge.

This challenge can be best illustrated by considering how a conventional DSM system is implemented [Li and Hudak 1989]. The global shared address space is divided into virtual memory pages. The local memory of each processor is used as a cache on the global shared address space. When a processor attempts to access a page of global virtual memory for which it does not have a copy, a page fault occurs. This page fault is handled by the DSM software, which retrieves a copy of the missing page from another node. If the access is a read, then the page becomes replicated in read-only mode. If the access is a write, then all other copies of the pages are invalidated. Throughout the rest of this article, the term *conventional DSM* [Li and Hudak 1989] refers to a DSM system that employs a page-based write-invalidate consistency protocol, such as the one just described.

The primary source of overhead in a conventional DSM system is the large amount of communication that is required to maintain consistency, or, put another way, to maintain the shared-memory abstraction. Ideally, the amount of communication for an application executing on a DSM system should be comparable to the amount of communication for the same application executing directly on the underlying message-passing system. Conventional DSM systems have found it difficult to achieve this goal because of restrictive memory consistency models and inflexible consistency protocols. The *false-sharing* problem is an example of this phenomenon. False sharing occurs when two threads on different machines concurrently update different shared data items that lie in the same virtual memory page. In conventional DSM systems, this false sharing can cause a page to "ping-pong" back and forth between different machines. In contrast, in a message-passing system, each thread would independently update its own copy of the data, without unnecessary communication. Some of these problems can be overcome by carefully restructuring the shared-memory programs to reflect the way that the DSM system operates. For example, one could decompose the shared data into small page-aligned pieces, or one could introduce new variables to reduce the amount of false sharing. However, this restructuring can be as tedious and difficult as using message passing directly.

1.2 Summary of Results

We present the following techniques for reducing the amount of communication needed for keeping the distributed memories consistent.

(1) *Software release consistency* is a software implementation of release consistency [Gharachorloo et al. 1990], specifically aimed at reducing the number of messages required to maintain consistency in a software DSM system. Roughly speaking, release consistency requires memory to be consistent only at specific synchronization points.

(2) *Multiple consistency protocols* are used to keep memory consistent in accordance with the observation that no single consistency protocol is best for all applications, or even for all data items in a single application [Bennett et al. 1990; Eggers and Katz 1988].

(3) *Write-shared protocols* address the problem of false sharing in DSM by allowing multiple processes to write concurrently into a shared page, with the updates being merged at the appropriate synchronization point, in accordance with the definition of release consistency.

(4) An *update-with-timeout* mechanism is in essence an update protocol that causes remote copies of shared data to be updated rather than invalidated. However, copies that are not referenced during the last timeout interval are deleted, eliminating the need for further updates and thus reducing the total amount of communication.

These techniques have been incorporated in the Munin DSM system. Munin has been implemented on a network of SUN-3/60 workstations running the V-System [Cheriton 1988]. The Munin programming interface is the same as that of conventional shared-memory parallel programming systems, except that it requires all synchronization to be visible to the runtime system and all shared variables to be declared as such and (optionally) annotated with the consistency protocol to be used. Other than that, Munin provides thread, synchronization, and data-sharing facilities such as those found in many shared-memory parallel programming systems.

To evaluate the benefits of these optimizations, we measured the performance of seven shared-memory parallel programs: Matrix Multiplication (MULT), Finite Differencing (DIFF), both a coarse-grained and a fine-grained version of the traveling salesman problem (TSP-C and TSP-F), Quicksort (QSORT), Fast Fourier Transform (FFT), and Gaussian Elimination with partial pivoting (GAUSS). Three versions of each program were written: a message-passing version, a Munin DSM version, and a conventional DSM version. The computational aspects of all three versions of each application were identical. The conventional DSM versions use a page-based write-invalidate protocol as described in Section 1.1.

Munin's performance is within 5% of message passing for MULT, DIFF, TSP-C, and FFT. For TSP-F, QSORT, and GAUSS, performance is within 29 to 33%. Detailed analysis of TSP-F and QSORT indicates that the addition of a function-shipping capability would bring their performance within 7% of the message-passing performance. Compared to a conventional DSM system, Munin achieves performance improvements ranging from a few percent for MULT to several hundred percent for FFT.

1.3 Outline of the Article

Section 2 describes the techniques for reducing consistency-related communication. Section 3 summarizes some aspects of the implementation that are relevant to the performance evaluation. Section 4 describes the applications used in the evaluation, as well as the experimental methodology. Section 5 contains an overview of the results, followed by a program-by-program comparison of the performance of the Munin, message-passing, and conventional DSM versions in Section 6. Section 7 attempts to isolate the benefits of the different techniques used to reduce consistency-related communication. Section 8 explores the additional performance benefits that could be achieved by

the use of function shipping. Related work is discussed in Section 9. We conclude in Section 10.

2. TECHNIQUES FOR REDUCING COMMUNICATION

This section describes the four techniques employed by the Munin DSM system to reduce consistency-related communication.

2.1 Software Release Consistency

Conventional DSM systems employ the *sequential-consistency* model [Lamport 1979] as the basis for their consistency protocols. Sequential consistency requires essentially that any update to shared data become visible to all other processors before the updating processor is allowed to issue another read or write to shared data [Li and Hudak 1989]. This requirement imposes severe restrictions on possible performance optimizations.

Among the various relaxed memory models that have been developed, we chose the release consistency model developed as part of the DASH project [Gharachorloo et al. 1990]. Release consistency exploits the fact that programmers use synchronization to separate accesses to shared variables by different threads. The system then only needs to guarantee that memory is consistent at select synchronization points. This ability to allow temporary, but harmless, inconsistencies is what gives release consistency its power. Consider, for example, a program where all access to shared data is enclosed in critical sections. Release consistency guarantees that when a thread successfully acquires the critical-section lock, it gains access to a version of shared data that includes all modifications made before the lock was last released. Similarly, for a program where all processes synchronize at a barrier, when a thread departs from the barrier, it is guaranteed to see all modifications made by all other threads before they reached the barrier. In general, if a program is free of data races, or, in other words, if there is synchronization between all conflicting shared-memory accesses, then the program generates the same results on a release-consistent memory system as it would on a sequentially consistent memory system [Gharachorloo et al. 1990]. Experience with release-consistent memories indicates that, because of the need to handle arbitrary thread preemption, most shared-memory parallel programs are free of data races even when written assuming a sequentially consistent memory [Carter et al. 1991; Gharachorloo 1991].

More formally, the following constraints on the memory subsystem ensure release consistency:

(1) Before an ordinary `read` or `write` is allowed to perform with respect to any other processor, all previous `acquire` accesses must be performed.

(2) Before a `release` access is allowed to perform with respect to any other processor, all previous `read` and `write` accesses must be performed.

(3) Synchronization accesses must be sequentially consistent with one another.

Lock acquires and lock releases map in the natural way onto `acquires` and `releases`. A barrier arrival is treated as a `release`, and a barrier departure

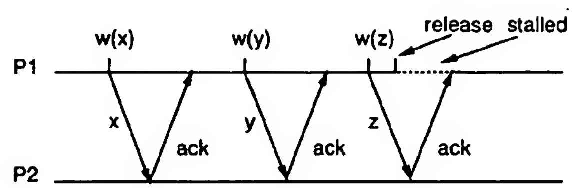

Fig. 1. Pipelining invalidations.

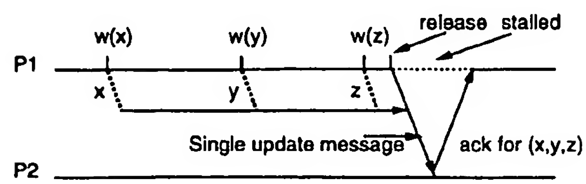

Fig. 2. Buffering and merging updates.

is treated as an acquire. Release consistency relaxes the constraints of sequential consistency in the following ways: (i) ordinary reads and writes can be buffered or pipelined between synchronization points, (ii) ordinary reads and writes following a release do not need to be delayed for the release to complete (i.e., a release only signals the state of *past* accesses to shared data), and (iii) an acquire access does not need to delay for previous ordinary reads and writes to complete (i.e., an acquire only controls the state of *future* accesses to shared data). The first point is the primary reason for release consistency's efficiency. Because ordinary reads and writes can be buffered or pipelined, a release-consistent memory can mask much of the communication required to keep shared data consistent.

2.1.1 *Buffered Update versus Pipelined Invalidate Release Consistency.* The hardware implementation of release consistency in DASH [Gharachorloo et al. 1990] *pipelines* invalidation messages caused by writes to shared data. This implementation is primarily geared toward masking the latency of writes, rather than reducing the number of messages sent. In a software DSM system, where the overhead of sending messages is very high, it is more important to reduce the frequency of communication than it is to mask latency by pipelining messages. For this reason, we developed an implementation of release consistency that *buffers* writes instead of pipelining them, as illustrated in Figures 1 and 2. These figures illustrate how writes to three shared variables (x, y, and z) within a critical section are handled by an implementation of release consistency that uses pipelining and an implementation that uses buffering, respectively. When a processor writes to several different replicated data items within a critical section, the pipelining scheme sends one message per write, while the buffering implementation buffers writes to shared data until the subsequent release, at which point it transmits the buffered writes. Ideally, the buffering implementation reduces the number of messages transmitted from one per write to one per critical section

when there is a single replica of the shared data. The dashed-line portion of the execution graph represents the delay that a processor experiences when releasing a lock. Because the buffering implementation delays all writes until the release point, it must transmit all buffered writes then, increasing the latency of releases. Nevertheless, the reduction in the number of messages far outweighs the effect of the higher release latencies.

Buffering and pipelining reduce the cost of writes, but have no effect on the cost of read misses. In software DSM systems, the cost of these read misses is very high, both in terms of communication and in terms of the length of time that a thread stalls before resuming after a read miss. The impact of read misses can be partially mitigated by using an update protocol. Update protocols based on sequential consistency may perform poorly because of the large amount of communication required to send update messages for every write. An update protocol based on release consistency can, however, buffer writes, which reduces substantially the amount of communication required.

2.2 Multiple Consistency Protocols

Most DSM systems employ a single protocol to maintain the consistency of all shared data. The specific protocol varies from system to system. For instance, Ivy [Li and Hudak 1989] supports a page-based write-invalidate protocol, whereas Emerald [Jul et al. 1988] uses object-oriented language support to handle shared-object invocations. Each of these systems, however, treats all shared data the same way. The use of a single protocol for all shared data leads to a situation where some programs can be handled effectively by a given DSM system, while others cannot, depending on the way in which shared data is accessed by the program. To understand how shared-memory programs access shared data characteristically, we studied the access behavior of a suite of shared-memory parallel programs. The results of this study [Bennett et al. 1990] and others [Eggers and Katz 1988; Weber and Gupta 1989] support the notion that using the flexibility of a software implementation to support *multiple consistency protocols* can improve the performance of DSM. They also suggest the types of access patterns that should be supported: *conventional*, *read-only*, *migratory*, *write-shared*, and *synchronization*.[1]

Conventional shared variables are replicated on demand and are kept consistent using an invalidation-based protocol that requires a writer to be the sole owner before it can modify the data. When a thread attempts to write to replicated data, a message is transmitted to invalidate all other copies of the data. The thread that generated the miss blocks until all invalidation messages are acknowledged. This single-owner consistency protocol is typical of what existing DSM systems provide [Dasgupta et al. 1990; Fleisch and

[1]The results of our original study [Bennett et al. 1990] indicated that there were eight basic access patterns (private, write-once, migratory, write-many, producer-consumer, result, read-mostly, and synchronization), but experience has made it clear that several of the protocols were redundant. Specifically, the *result* and *producer-consumer* access patterns were subcases of the *write-shared* access pattern.

Popek 1989; Li and Hudak 1989], and is what we use exclusively to represent a conventional DSM system in our performance evaluation.

Once *read-only* data has been initialized, no further updates occur. Thus, the consistency protocol simply consists of replication on demand. A runtime error is generated if a thread attempts to write to read-only data.

Migratory data is accessed multiple times by a single thread, including one or more writes, before another thread accesses the data [Bennett et al. 1990; Weber and Gupta 1989]. This access pattern is typical of shared data that is accessed only inside a critical section or via a work queue. The consistency protocol for migratory data propagates the data to the next thread that accesses the data, provides the thread with read *and* write access (even if the first access is a read), and invalidates the original copy. This protocol avoids a write miss and a message to invalidate the old copy when the new thread first modifies the data.

Write-shared variables are frequently written by multiple threads concurrently, without intervening synchronization to order the accesses, because the programmer knows that each thread reads from and writes to different portions of the data. Because of the way that the data is laid out in memory, access to write-shared data suffers from the effects of *false sharing* if the DSM system attempts to keep these different portions of the data consistent at all times. This protocol is discussed in more detail in Section 2.3.

We support three types of *synchronization* variables: locks, barriers, and condition variables. Because synchronization variables are accessed in a fundamentally different way than normal data objects, it is important that synchronization *not* be provided through shared memory, but rather via a suite of synchronization library routines or a similarly specialized implementation. Doing so reduces the number of messages required to implement synchronization, especially compared to conventional spinlock algorithms, and thereby reduces the amount of time that threads spend blocked at synchronization points.

2.3 Write-Shared Protocol

The write-shared protocol is designed specifically to mitigate the effect of false sharing, as discussed in Sections 1 and 2.2. False sharing is a particularly serious problem for DSM systems for two reasons: the consistency units are large, so false sharing is very common; and the latencies associated with detecting modifications and communicating are large, so unnecessary faults and messages are particularly expensive. The write-shared protocol allows concurrent writers and buffers writes until synchronization requires their propagation (see Figure 2).

In order to record the modifications to *write-shared* data, the DSM system initially write-protects the virtual memory pages containing the data. When a processor first writes to a page of write-shared data, the DSM software makes a copy of the page (a *twin*) and queues a record for the page in the delayed-update queue (DUQ), as shown in Figure 3. The DSM then removes write protection on the shared data so that further writes can occur without any DSM intervention.

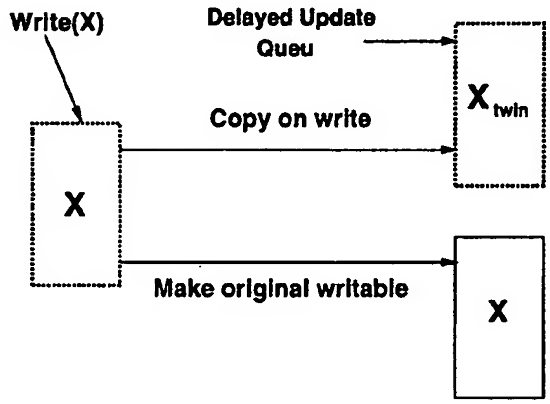

Fig. 3. Write-shared protocol: Creating twins.

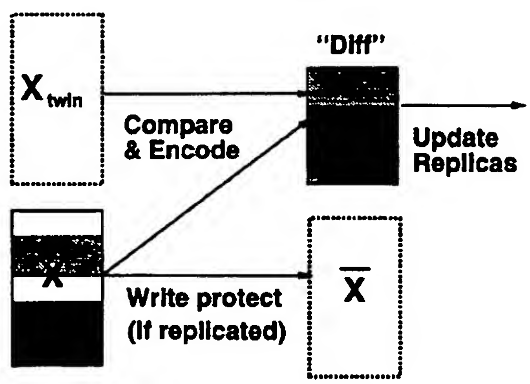

Fig. 4. Write-shared protocol: Sending out *diffs*.

At release time, the DSM system performs a word-by-word comparison of the page and its twin, and run-length encodes the results of this *diff* into the space allocated for the twin (see Figure 4). Each encoded update consists of a count of identical words, the number of differing words that follow, and the data associated with those differing words. Each node that has a copy of a shared object that has been modified is sent a list of the available updates. Nodes receiving update notifications request the updates they require,[2] decode them, and merge the changes into their versions of the shared data. A

[2] If all the encoded updates fit into a single packet, they are sent directly in place of the list of available updates, thus eliminating unnecessary communication in the event that only a small amount of shared data has been modified.

runtime switch allows this comparison to be performed at the byte level, as opposed to the word level, if the data is more finely shared.

Another runtime switch can be set to check for conflicting updates to write-shared data. If this switch is set, then, when a *diff* arrives at a processor that has a dirty copy of the page, the DSM system checks whether any of the updates in the *diff* conflict with any of the local updates, and, if so, signals an error. The ability to detect conflicting updates allows Munin to support dynamic data race detection.

2.4 Update Timeout Mechanism

The performance of update protocols suffers from the fact that updates to a particular data item are propagated to all its replicas, including those that are no longer being used. This problem is particularly severe in DSM systems, because the main memories of the nodes in which the replicas are kept are very large, and it takes a long time before a page gets replaced, if at all. Without special provisions, updates to these *stale* replicas can lead to a large number of unnecessary consistency messages, resulting in poor performance. This effect is one reason that existing commercial multiprocessors use invalidation-based protocols. We address this problem with a timeout algorithm similar to the competitive snoopy caching algorithm devised by Karlin et al. [1986]. The goal of the update timeout mechanism is to invalidate replicas of a cached variable that have not been accessed recently upon receipt of an update.

Munin's update timeout mechanism is implemented as follows. When receiving an update for a page for which no twin exists locally, the page is mapped such that it can be accessed only in supervisor mode, and the time of receipt of this update is recorded. A local access causes a fault, as a result of which protection is removed and the timestamp is reset. If the page is still in supervisor mode when another update arrives (meaning it has not been accessed locally since the first update), and a certain time window δ has expired (50 milliseconds in the prototype), then the page is invalidated, and a negative acknowledgment is sent to the originator of the update, causing it to send updates to this processor no more. In addition to avoiding unnecessary updates, the update timeout mechanism often reduces the number of messages sent in conjunction with updates to stale data. When a node receives an update message from another node that includes stale updates, the recipient node does not request the actual modifications associated with the shared data it is no longer caching. Thus, unless all the updates described in the update message are to stale data, no extra work is performed to process the stale updates other than the small amount of processing necessary to note that the updates are not needed. If all the updates are to stale data, the overhead is only a single packet exchange.

The use of update timeouts results in a hybrid update-invalidate protocol that allows Munin to gain the benefits of an update mechanism—i.e., the reduction in the number of read misses and subsequent high-latency (idle) reloads—while at the same time retaining the superior scalability of an

invalidation protocol by limiting the extent to which stale copies of particular pages are updated.

3. THE MUNIN DSM PROTOTYPE

The techniques described in Section 2 have been implemented in the Munin DSM system [Carter et al. 1991]. Munin was evaluated on a network of SUN-3/60 workstations running the V-System [Cheriton 1988] connected via an isolated 10MB-per-second Ethernet. This section provides a brief overview of aspects of the implementation of Munin that are relevant to its evaluation. A more-detailed description of the Munin prototype appears elsewhere [Carter 1993].

3.1 Writing A Munin Program

Munin programmers write parallel programs using threads, as they would on many shared-memory multiprocessors. Synchronization is supported by library routines for the manipulation of locks, barriers, and condition variables. All of the current applications were written in C.

Currently, Munin supports only statically allocated shared variables, although support for dynamically allocated shared data could be added easily. The programmer annotates the declaration of shared variables to specify what protocol to use to keep shared data consistent: for example, `shared {protocol} ⟨C_type⟩ ⟨variable_name⟩`. The keyword `shared` is required to specify that a variable will be shared among processes, although the `protocol` can be omitted. If the protocol is omitted, the conventional protocol is used. Incorrect protocol annotations may result in inefficient performance, or in runtime errors that are detected by the Munin runtime system, but not in incorrect results. All of the shared data in the test programs was fully annotated.

3.2 Compiling and Linking a Munin Program

A preprocessor filters the source code in search of shared-variable declarations. For each such declaration, the preprocessor removes the Munin-specific `shared {protocol}` portion and adds an entry to an auxiliary file. After preprocessing, the source file is compiled with the regular compiler. The Munin linker reads the auxiliary file and relocates the shared variables to a shared segment. By default, the linker places each shared variable on a separate page. In addition, the Munin linker appends to the executable a shared-segment symbol table that describes the layout of the shared memory and the protocols to be used for the shared data. These additions to Munin executables had a negligible impact on program size or startup costs.

3.3 Runtime Overview

Figure 5 illustrates the organization of a Munin program during runtime. On each participating node, the Munin library is linked into the same address space as the user program, and thus can access user data directly. The two major data structures used by the Munin runtime system are the *delayed-*

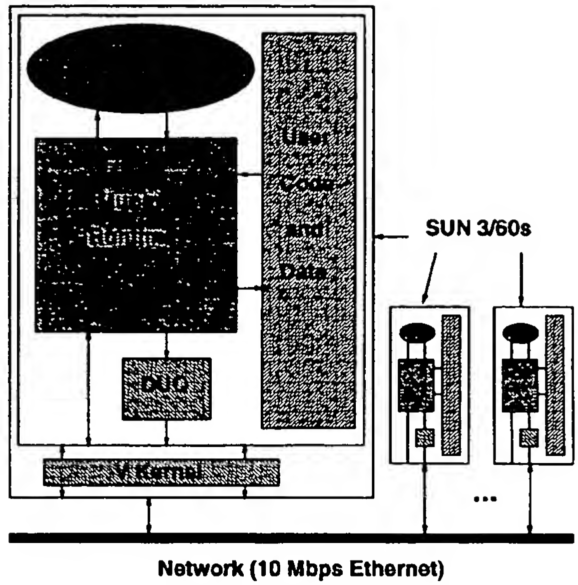

Fig. 5. Munin runtime organization.

update queue (see Section 2) and the *object directory*, which maintains the state of the shared data being used by local user threads. A Munin system thread installs itself as the page fault handler for the Munin program. As a result, the underlying V kernel [Cheriton 1988] forwards all memory exceptions to this thread. The Munin thread also interacts with the V kernel to communicate with the other Munin nodes over the network and to manipulate the virtual memory system as part of maintaining the consistency of shared memory. The prototype uses no features of V for which equivalent features are not commonly available on other platforms (e.g., UNIX or Mach). In addition, we avoided using features that we believed might not be common on future workstation clusters, such as reference bits in the page table or a multicast capability on the network. For the update timeout mechanism, references are detected by mapping write-shared pages to supervisor mode so that the first reference to a page after it is updated results in a page fault. We thus maintain a reference bit and timestamp for each page without requiring hardware-supported reference bits. Although the prototype runs on a collection of workstations connected via an Ethernet, the multicast capability of Ethernet was not used so that our results could be generalized to platforms without hardware multicast.

3.4 The Object Directory

On each node, the Munin runtime system maintains a page-level object directory containing information on the state of each data item in the global

shared memory, as shown in Figure 5. All shared variables on the same physical page are treated as a single object. Variables that are larger than a page, e.g., a large array, are treated as a number of independent page-sized objects. Munin uses variables rather than pages as the basic unit of granularity because this reflects better the way data is used and reduces the amount of false sharing between unrelated variables [Bennett et al. 1990].

Munin's strategies for maintaining the object directory are designed to reduce the number of messages required to maintain the distributed object directory. First, in keeping with the goal of avoiding centralized algorithms, Munin distributes the state information associated with write-shared data across the nodes that contain cached copies of the data. In many cases, this elimination of the notion of a static "owner" of data allows nodes to respond to requests completely locally. This is done by allowing directory entries to be inconsistent at times. This approach also allows Munin to exploit locality of reference when maintaining directory information, because the need to maintain a single, consistent directory entry, as has been proposed for most scalable shared-memory multiprocessors, is eliminated. Second, Munin implements a dynamic ownership protocol to distribute the task of data ownership across the nodes that use the data. In general, when a shared data item is not owned by the local node, the information in the local directory entry acts as a "hint" to reduce the overhead of performing consistency operations.

3.5 Synchronization Support

Synchronization objects are accessed in a fundamentally different way than ordinary data [Bennett et al. 1990]. Thus Munin provides efficient implementations of locks, barriers, and condition variables that directly use V's communication primitives rather than synchronizing through shared memory. More-elaborate synchronization mechanisms, such as monitors and atomic integers, can be built using these basic mechanisms. Each Munin node maintains a synchronization object directory, analogous to the data object directory, containing state information for the synchronization data. All of Munin's synchronization primitives cause the local delayed-update queue to be purged on a "release."

3.5.1 *Locks.* Munin employs a queue-based implementation of locks similar to existing implementations on shared-memory multiprocessors. This allows a thread to request ownership of a lock and block awaiting a reply, without repeated queries. The system associates an ownership "token" and a distributed queue with each lock. A *probable-owner* mechanism is used to locate the token or the end of the queue associated with the lock. The token migrates to nodes as they become owners, so no single node is responsible for maintaining the state of a given lock. This approach has the same benefits in terms of exploiting locality of reference, removing central bottlenecks, and reducing communication as does Munin's distributed data ownership protocol. A frequent situation in which this scheme works to particular advantage is when a thread attempts to reacquire a lock for which it was the last owner [Bennett et al. 1990]. In this case, the thread finds the associated token to be

available locally and is thus able to acquire the lock immediately (without any message overhead). Similarly, if a small subset of threads continuously reuses the same lock, they communicate only with one another.

When the lock ownership token is unavailable locally, a message is sent along the probable-owner chain to the last lock holder. If the lock is free (the token is available), the last lock holder forwards the token to the requester, which acquires the lock and continues executing. Otherwise, the thread that was at the end of the queue stores the locking thread's identity into a local data structure without replying. Each enqueued thread knows the identity of the thread that follows it on the queue, if any, so when a thread releases a lock and the associated queue is nonempty, lock ownership is forwarded directly to the next thread in the queue after all delayed updates are flushed in accordance with the requirements of release consistency.

3.5.2 *Barriers.* Barriers are used to synchronize multiple threads simultaneously. When a barrier is created, the user specifies the number of threads that must reach the barrier before it is lowered. When a thread wishes to wait at a barrier, it flushes any delayed updates, sends a message to the barrier manager thread (a well-known thread located on the root node, from where the Munin program was invoked), and awaits a response. When all the threads have arrived at the barrier, the barrier manager replies to each waiting thread to let it resume. We considered using a distributed barrier mechanism similar to those designed for scalable multiprocessor systems, but for the small size of the prototype implementation, a simple centralized scheme was more practical and efficient. Unlike locks, which are point-to-point and which exhibit a high degree of locality that makes it beneficial to migrate ownership, barriers are most often used to synchronize all the user threads in the program. In this case, locality of reference cannot be exploited, because single threads or small subsets of threads do not tend to access the barrier without intervening accesses by other threads. Thus, until the single barrier manager becomes a bottleneck, there is no reason to distribute barrier ownership.

3.5.3 *Condition Variables.* Munin's condition variables are essentially binary semaphores that also support a broadcast wakeup capability. Unlike locks, condition variables give threads the capability to synchronize indirectly. Any thread can perform a signal operation, but the lock protocol allows only the lock owner to release the lock. Although it is possible to build this kind of mechanism using locks, we found it convenient to include condition variables as a primitive. In accordance with the requirements of the release consistency model, delayed modifications are flushed before the signal or broadcast message is forwarded to the condition manager thread.

4. EVALUATION

4.1 Application Programs

Seven application programs were used in the evaluation. Three different versions of each application were written: a Munin DSM version, a conven-

Table I. Programs and Problem Sizes Used

Program	Problem Size
MULT	400-by-400 square matrices
DIFF	512-by-512 square matrices
TSP-C	18 cities, recurse when < 13
TSP-F	18 cities, recurse when < 12
QSORT	256K items, recurse when < 1024
FFT	32K elements
GAUSS	256-by-256 square matrices

tional DSM version that used the conventional protocol for a sequentially consistent memory, and a message-passing version. Great care was taken to ensure that the "inner loops" of each computation, the problem decomposition, and the major data structures for each version were identical. Except where noted, all array elements are double-precision floating-point numbers. Both the DSM system and the message-passing programs used V's standard communication mechanisms.

The DSM programs were originally written for a shared-memory multiprocessor (a Sequent Symmetry). Our results may therefore be viewed as an indication of the possibility of "porting" shared-memory programs to software DSM systems, but it should be recognized that better results may be obtained by tuning the programs to a particular DSM environment. Table I summarizes the seven application programs and problem sizes. An effort was made to select a suite of programs that would represent a relatively wide spectrum of shared-memory parallel programs, varying in their parallelization techniques, granularity, degree and nature of sharing, and locality of shared-data references. Matrix Multiply (MULT), Finite Differencing (DIFF), and Gaussian Elimination with partial pivoting (GAUSS) are numeric problems that statically distribute the data across the threads. MULT, DIFF, and GAUSS exhibit increasing degrees of sharing. FFT reallocates the data across threads dynamically and exhibits an extremely high degree of sharing. The traveling salesman problem (TSP) and Quicksort (QSORT) programs use the task queue model to allocate work dynamically to different threads. The granularity for TSP was varied (TSP-C and TSP-F access data at a coarse and fine grain, respectively). QSORT exhibits a high degree of false sharing in the array to be sorted. Small to moderate problem sizes were chosen so that the uniprocessor running times would be in the range of hundreds of seconds, and the 16-processor running times would be on the order of tens of seconds. The uniprocessor running times represent sequential implementations of the programs with all synchronization and communication removed.

4.2 Experimental Methodology

For all three versions of each program, a sequential initialization routine is executed on the root node. Then the appropriate number of additional nodes is created, which for the DSM versions gives each node a copy of the nonshared data. The nonroot nodes initialize themselves, and then synchro-

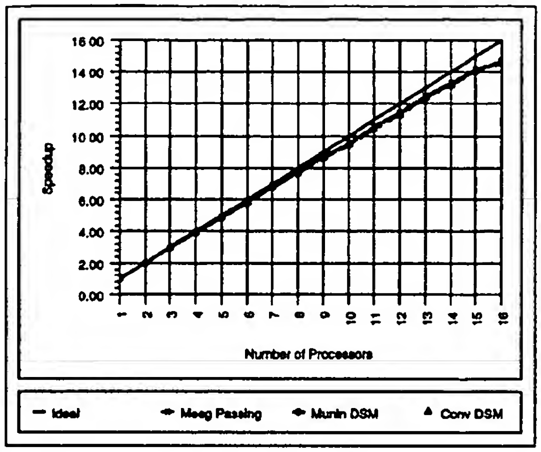

Fig. 6. Matrix Multiplication (MULT).

nize with the root node by waiting at a barrier for the DSM versions and via an explicit message in the message-passing versions. For the DSM versions, after the user thread on the root node has created the required worker threads on each node, it reads the clock to get the initial value and then waits at the barrier, which causes the computation to begin. For the message-passing versions, the root thread waits until it has received the "initialization complete" message from all the worker threads. It then reads the initial clock value and sends a message to each worker to start computation. At this point, the workers read their inputs, via page faults for the DSM versions or via request messages for the message-passing versions. Once all the workers have completed, the root thread reads its clock again and calculates the total elapsed computation time.

In addition to execution times, the Munin runtime system gathers statistics on the number of faults, the amount of data transferred, and the amount of time stalled while performing various consistency operations. The message-passing kernel collects similar data. Selected portions of these statistics are used throughout the analysis to highlight the reasons for observed performance differences between the different versions of the programs.

5. OVERVIEW OF RESULTS

The main results we report are the speedup of the various versions of the parallel programs over the sequential version, measured for 2 to 16 processors. Figures 6 through 12 show the speedup for each of the application programs as a function of the number of processors. Table II shows the speedup achieved on 16 processors for the three versions of each application. The percentages in parentheses represent the percentage of message-passing speedup achieved by Munin, and the percentage of both message-passing and Munin's speedup achieved by the conventional DSM implementation. Tables

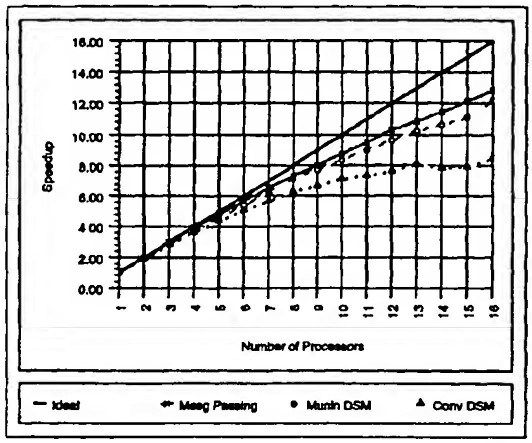

Fig. 7. Finite Differencing (DIFF).

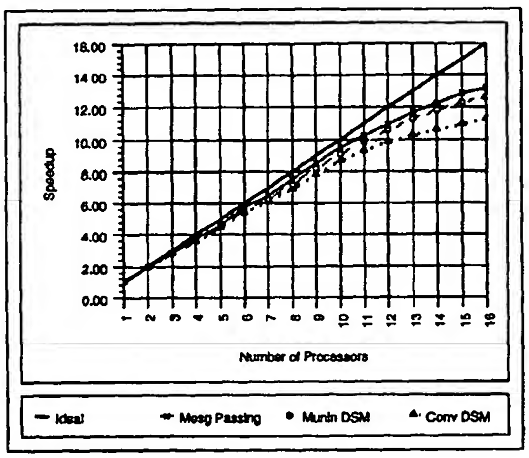

Fig. 8. Coarse-grained traveling salesman problem (TSP-C).

III and IV show the amount of communication required during execution of the programs on 16 processors, both in terms of number of messages and kilobytes of data transmitted.

For MULT, DIFF, TSP-C, and FFT, the Munin versions achieved over 95% of the speedup of their hand-coded message-passing equivalents, whereas for TSP-F, QSORT, and GAUSS the Munin programs achieved between 67 and 71%. For the programs with large-grain sharing (MULT and TSP-C), the conventional versions achieved 99 and 90%, respectively, of the speedup of their Munin counterparts. For DIFF, TSP-F, QSORT, and GAUSS the perfor-

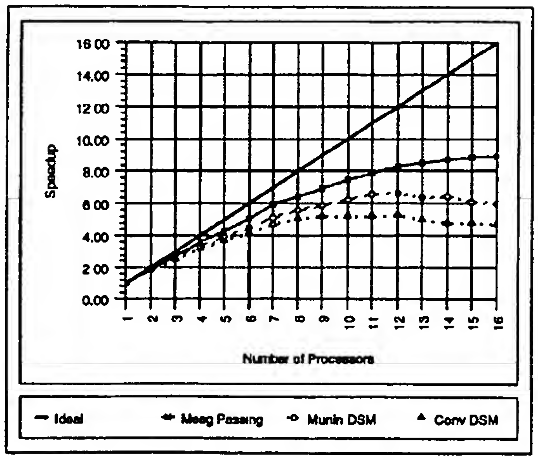

Fig. 9. Fine-grained traveling salesman problem (TSP-F).

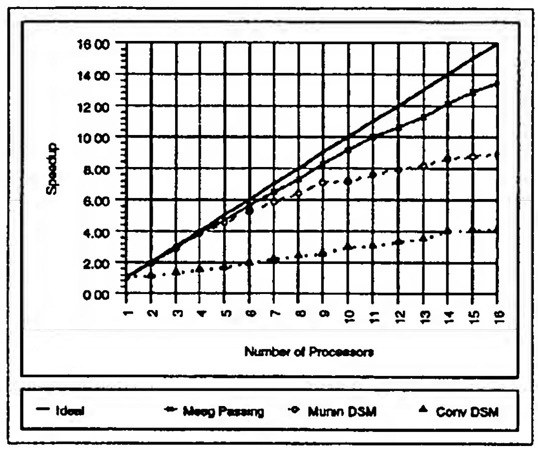

Fig. 10. Quicksort (QSORT).

mance of the conventional versions was reduced to 46–80% of Munin. For FFT, there was so much false sharing that the conventional version slowed down by a factor of ten when run on more than one processor.

6. DETAILED ANALYSIS

In this section we analyze in detail, on a per-program basis, the reasons for the performance differences among the various versions of each program. Unless otherwise noted, the numbers in this section pertain to the 16-processor execution.

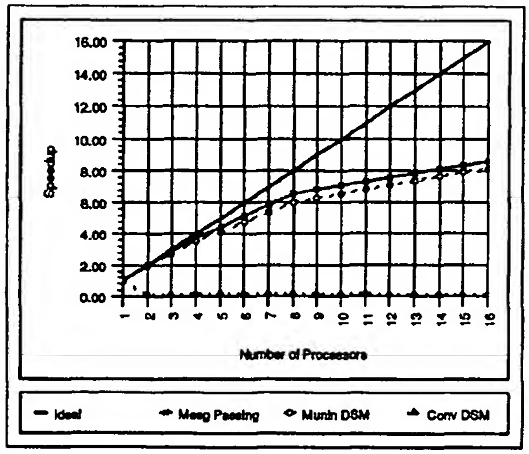

Fig. 11. Fast Fourier Transform (FFT).

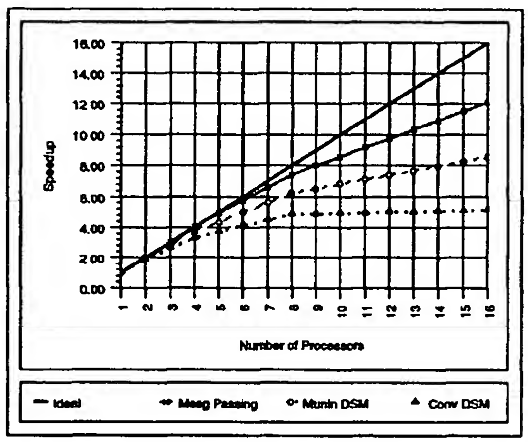

Fig. 12. Gaussian elimination with partial pivoting (GAUSS).

Table II. Speedups Achieved (16 Processors)

Program	Message Passing	Munin DSM	Conventional DSM
MULT	14.7	14.6 (100%)	14.5 (99%, 99%)
DIFF	12.8	12.3 (96%)	8.4 (66%, 68%)
TSP-C	13.2	12.6 (96%)	11.3 (86%, 90%)
TSP-F	8.9	6.0 (67%)	4.7 (53%, 80%)
QSORT	13.4	8.9 (67%)	4.1 (31%, 46%)
FFT	8.6	8.2 (95%)	0.1 (0%, 0%)
GAUSS	12.1	8.6 (71%)	5.1 (42%, 59%)

Table III. Number of Messages for 16-Processor Execution

Program	Message Passing	Munin	Conventional
MULT	672	1567	1490
DIFF	14164	14646	35486
TSP-C	902	7870	7940
TSP-F	919	9776	10194
QSORT	667	31866	129428
FFT	9225	15322	1594952
GAUSS	14768	26034	32349

Table IV. Amount of Data (in Kilobytes) for 16-Processor Execution

Program	Message Passing	Munin	Conventional
MULT	640	1384	1327
DIFF	8294	3645	26534
TSP-C	68	4163	4770
TSP-F	68	4989	5963
QSORT	524	14565	101007
FFT	9339	11621	1336317
GAUSS	4995	5526	7388

6.1 Matrix Multiply

Program Description. The problem is to multiply two $N \times N$ input arrays and put the result in an $N \times N$ output array. Matrix Multiply is parallelized by giving each worker thread a number of contiguous rows of the output array to compute. After each worker thread has terminated, the root thread reads in the result array and terminates.

The DSM versions use a barrier to signal completion; each worker thread in the message-passing version sends its result rows to the master when they have been computed. The Munin version declares the input arrays as `read_only` and the output array as `write_shared`.

Analysis. Matrix multiplication is almost completely compute bound. As a result, the three versions achieved almost identical speedups (14.5 for conventional DSM, 14.6 for Munin, and 14.7 for message passing). In all cases, the cumulative computation time is roughly 900 seconds, although the cumulative communication time is roughly four seconds. Both the Munin and the conventional DSM versions perform approximately twice as much communication as the message-passing version, because the DSM worker threads fault in the empty result array at the beginning of the computation, whereas the message-passing worker threads simply initialize their portion of the result array in place. Also, in Munin, when a thread arrives at the final barrier, it updates any copies of a page in the result matrix that are cached by neighboring nodes due to false sharing. This results in the Munin version performing more communication than the conventional version. The Munin version still outperforms the conventional version because the extra commu-

nication is largely overlapped with computation, whereas the read misses experienced by the conventional version cause processors to stall. Nevertheless, compared to the overall execution time, the time spent communicating is minor, so both the conventional and Munin versions exhibit near-linear speedup.

6.2 Finite Differencing

Program Description. During each iteration of the finite-differencing algorithm, all elements of a matrix are updated to the average of their nearest neighbors (above, below, left, and right). To avoid overwriting the old value of a matrix element before it is used, an iteration is split in two half-iterations. In the first half-iteration, the program uses a scratch array to compute the new values. In the second, it copies the scratch array back to the main matrix.

Each thread is assigned a number of contiguous rows to compute. The algorithm requires only those elements that lie directly along the boundary between two threads' subarrays to be communicated at the end of each iteration. In the Munin version, the matrix is declared as `write_shared`. In the DSM versions, the programmer is *not* required to specify the data partitioning to the runtime system—it is inferred at runtime based on the observed access pattern. After each half-iteration, the DSM worker threads synchronize by waiting at a barrier. The message-passing workers exchange results directly between neighboring nodes after each iteration.

Analysis. DIFF has a much smaller computation-to-communication ratio than MULT (see Tables III and IV), but the Munin version still performs within 5% of the message-passing version (a speedup of 12.3 for Munin versus 12.8 for message passing). The reason for Munin's good performance is its use of software release consistency and the write-shared protocol. Together, these techniques result in the underlying communications patterns for the Munin version and the message-passing version being nearly identical. When each thread first accesses a page of shared data, it gets a copy of the page. Thus, at the end of the first half-iteration, each node has a read-write copy of any pages for which it has the only copy, and a read-only copy of any pages that lie along a boundary. During the second half-iteration, during which each thread copies the new values from the scratch array to the shared array, each node creates a *diff* of its shared pages. When a thread arrives at the barrier after this half-iteration, it sends the *diff* directly to the appropriate neighbors before sending the barrier message to the barrier master. These *diffs* include all the modified data on each boundary page, and not just the edge elements. Since the shared pages are still shared even after they are purged, they are write-protected again, so subsequent writes will be detected. For subsequent iterations, each node experiences a protection violation only on the boundary pages, and then only performs local operations (creating twins), except when exchanging the results. Thus, the data motion in the Munin version of DIFF is essentially identical to the message-passing implementation—communication only occurs at the end of each iteration,

and only neighboring nodes exchange results. The only overhead comes from fault handling, and from copying, encoding, and decoding the shared portions of the matrix.

As an aside, a curious phenomenon can be seen in Table IV. The Munin version of DIFF transmits less data than the message-passing version. This is a result of the fact that Munin only transmits the words that have been modified during each iteration, whereas the message-passing version ships the entire edge row. During the early iterations, many of the edge values have not been modified yet, and thus Munin does not transmit any new values for them. In practice, this extra transmitted data had a negligible effect on the running times. Rather, Munin's good performance derived from the fact that it transmits data only during synchronization and suffers no read misses (after the first iteration).

The conventional DSM version of DIFF achieved a speedup of only 8.4, compared to 12.3 for Munin. The conventional version suffers from (1) frequent read faults and reloads as a result of the invalidation protocol and (2) blocking on write faults as a result of sequential consistency. The Munin version of DIFF creates and transmits *diffs* at the end of each iteration, which results in shared data being present before it is accessed during the next iteration. This eliminates read misses and reloads on the next iteration. In contrast, the conventional DSM implementation invalidates and reloads every shared page in its entirety on each iteration. In addition, write faults can be handled completely locally in Munin if the data are already present, which is the case for all but the first iteration. The local node simply makes a twin of the data. The conventional DSM implementation sends an invalidation message and waits for a response. The tradeoff is that synchronization under Munin is slowed down because memory needs to be made consistent before the synchronization operation can complete. However, the total time that the Munin worker threads spend blocked while waiting for memory to be made consistent (71.5 seconds) is far less than the time spent invalidating and reloading the data in the conventional version (a total of 356.1 seconds). The time spent invalidating and reloading has a serious impact on execution time (356.1 seconds of a total execution time of 662.1 seconds).

6.3 Traveling Salesman Problem

Program Description. The traveling salesman problem (TSP) takes as its input an array representing the distances between cities on a salesman's route and computes the minimum-length "tour" passing through each city exactly once. A *tour queue* maintains a number of partially evaluated tours. If the number of nodes remaining to complete the tour is below a threshold, 12 for TSP-F and 13 for TSP-C, the remainder of the tour is evaluated sequentially. If the number of nodes remaining is above this threshold, the partial tour is expanded by one node, and the new partial tours are entered on the tour queue. When a partial tour is removed from the queue, a lower bound on the remaining part of the tour is computed, and the tour is rejected if the sum of the current length and the lower bound is higher than the

current best tour. This check is also performed before a potential new subtour is put on the task queue. The tour queue is a *priority queue* that orders the remaining subtours in the inverse order of a lower bound of their total length. Thus the "most-promising" subtours are evaluated first, which tends to prune uninteresting subtours more quickly. The major shared data structures of TSP are the current shortest tour and its length, an array of structures that represent partially evaluated tours, a priority queue that contains indices into the tour array of partially evaluated tours, and a stack of indices of unused tour array entries. TSP-C and TSP-F differ only in the problem granularity. TSP-C solves subtours of length 13 or less sequentially, whereas TSP-F solves subtours of length 12 or less sequentially. Depending on the particular input data set, the computation-to-communication ratio of TSP-C can be as much as ten times higher than that of TSP-F.

In the DSM versions, locks protect the priority queue, the current shortest tour, and its length. A condition variable is used to signal when there is work to be performed. Worker threads acquire the lock and continue to remove partial tours from the queue until a "promising" tour has been found that can be expanded sequentially, at which time the lock is released. In Munin, the priority queue and the stack of unused tours are declared `migratory`, and the other shared data structures are declared `write_shared`. For the message-passing version, the *master* maintains a central priority queue that contains the indices of subtours to be solved. The *slaves* send request messages to the master, which responds either with a subtour to be solved sequentially, or an indication that there is no more work. Workers tell the master when they find a new global minimum, and the master is responsible for propagating it.

Analysis (Coarse-Grain TSP). The Munin version achieved a speedup of 12.6, within 5% of the 13.2 achieved by the message-passing version. TSP-C is rather compute bound: under 30 seconds of communication for the Munin version compared to a total execution time of 880 seconds. The performance difference between the message-passing version and the Munin version comes from the cost of accessing the priority queue. In Munin, each time a thread tries to remove a tour from the queue, the queue data structure needs to be shipped to that thread. This behavior had two adverse effects on performance: worker threads cumulatively spent 62 seconds waiting on the task queue lock, and the Munin version shipped 4MB of data, compared to only 900KB in the message-passing version.

The difference in performance between the Munin and conventional DSM versions of TSP-C (a speedup of 12.6 for Munin versus 11.3 for conventional DSM) stems from the use of a migratory protocol for the task queue and the use of an update, instead of an invalidate, protocol for the minimum tour length. The slightly higher overhead caused by loading and invalidating, rather than simply migrating, the task queue had the effect of causing more processors to idle themselves waiting for work. This was because access to the task queue was the primary bottleneck (a total of 94 seconds for the conventional version versus only 62 in the Munin version). The minimum tour

length is an example of a shared data item for which an update protocol is better than an invalidate protocol, because it is read much more frequently than it is written. With the conventional protocol running on N processors, a thread that needs to update the minimum tour length typically sends $N - 1$ invalidations and then waits for $N - 1$ acknowledgments. All other threads in turn incur an access miss, and its associated latency, to obtain a new copy of the minimum tour length.

Analysis (Fine-Grain TSP). The Munin version of TSP-F achieved a speedup of 6.0, 33% less than the 8.9 speedup achieved by the message-passing version. The reasons for the reduction in performance are the same as for TSP-C, but their relative importance is increased. In TSP-F, worker threads spent a cumulative 360 seconds waiting for the priority queue, and a total of 210 seconds performing useful computation. In addition, 9.2MB of data were transmitted in the Munin version, compared to only 920KB for the message-passing version. Similar arguments apply for the conventional DSM version, resulting in a speedup of only 4.7.

6.4 Quicksort

Program Description. Quicksort (QSORT) is a recursive sorting algorithm that operates by repeatedly partitioning unsorted input lists into unsorted sublists such that all the elements in one of the sublists are strictly greater than the elements of the other. The Quicksort algorithm is then recursively invoked on the two unsorted sublists. The base case of the recursion occurs when the lists are sufficiently small (1KB in our case), at which time they are sorted sequentially.

Quicksort is parallelized using a work queue that contains descriptors of unsorted sublists, from which worker threads continuously remove unsorted lists. In the DSM versions of QSORT, the major data structures are the array to be sorted, a task queue that contains range indices of unsorted subarrays, and a count of the number of worker threads blocked waiting for work. Like TSP, the task queue is declared to be `migratory`, and the array being sorted is declared to be `write_shared`. A lock protects the queue, and a condition variable is used to signal the presence of work to be performed. QSORT differs from TSP in that when QSORT releases control of the task queue, it may need to subdivide the work further by partitioning the subarray and placing the new subarrays back into the task queue. In contrast, TSP workers never relinquish control of the task queue until they have removed a subtour that can be solved sequentially. Therefore, the task queue in QSORT is accessed more frequently per unit of computation. Offsetting this is the fact that the threads in TSP hold the lock protecting the priority queue for a longer time as they perform the expansion.

For the message-passing version of QSORT, the master maintains the work queue. The slaves send request messages to the master, which responds either with the sublist to be sorted sequentially or an indication that there is no more work. Along with the requests, the slaves ship the sorted results from their previous request, if any.

Analysis. The Munin version of QSORT achieves only 67% of the speedup of the message-passing version (8.9 versus 13.4). As with TSP-C and TSP-F, most of Munin's overhead comes from shipping the work queue each time a node tries to perform a queue insertion or deletion. Compounding this problem is the fact that the threads do not retain sole ownership of the work queue while subdividing the work into pieces sufficiently small to solve directly, so they repeatedly need to reacquire the task queue and partition their subarray until it contains at most 1024 elements. As a result, the threads spent a cumulative 842 seconds waiting on the task queue lock, out of a total execution time of 2160 seconds. Furthermore, the Munin version transmitted 23MB of data, compared to 520KB for the message-passing implementation.

For the conventional DSM version, speedup drops to 4.1. In addition to the cost of invalidating and reloading the task queue, rather than simply migrating it, the difference in performance between the conventional DSM version and the Munin version is primarily due to the presence of false sharing when two threads attempt simultaneously to sort subarrays that reside on the same page. As a result, communication goes from 23MB in about 30,000 messages for the Munin version to 110MB in 231,000 messages for the conventional version.

6.5 Fast Fourier Transform

Program Description. The Fast Fourier Transform (FFT) program used in the evaluation is based on the Cooley-Tukey Radix 2 Decimation in Time algorithm. It recursively subdivides the problem into its even and odd components, until the input is of length 2. For this base case, the output is an elementary function known as a *butterfly*, a linear combination of its inputs. For an input array of size N, the FFT algorithm requires $\log_2 N$ passes. On pass K, the width of each butterfly is $N2^{-(K+1)}$. Thus, for the first pass, the width of the bufferfly is $N/2$, and on each subsequent iteration the width of each butterfly halves. By starting with the wide butterflies, the result array is a permutation of the desired value, but this is rectified with an $O(N)$ cleanup phase.

If P processors are used to solve an N-point FFT, where P is power of 2, then a reasonable initial decomposition of the work allows processor p to work with $x[p]$, $x[p+P]$, $x[p+2P], \ldots, x[p+N-P]$. This allows all processors to perform the first $\log_2 N - \log_2 P$ passes without any interprocessor communication. Before executing the last $\log_2 P$ iterations, the processors exchange data and reallocate themselves to different (contiguous) subarrays.

Both the DSM and message-passing programs are parallelized by dynamically allocating threads to data as previously described. The array on which the FFT is being performed is declared to be `write_shared` in the Munin version. By carefully allocating processors to data as shown, it is possible to only reallocate the processors and exchange data at the end of the first $\log_2 N - \log_2 P$ phases. The DSM programs use a barrier to synchronize at

this point. The DSM system automatically reallocates the data on demand. The message-passing version manually encodes and shuffles the data, using a master process to collect and redistribute all changes. This manual redistribution made the message-passing version much harder to write than the DSM versions. The processor reallocation is built into the algorithm itself.

Analysis. The FFT algorithm used has a very high degree of sharing, which results in it being bus bandwidth limited to a speedup of approximately 10 on a 20-processor, single-bus multiprocessor like the Sequent Symmetry. Because of the way that the data is distributed, every page is referenced (and modified) by every thread during the first $\log_2 N - \log_2 P$ iterations, the worst possible behavior for any DSM system. The conventional DSM version *slows down* by a factor of 10 for two or more processors, whereas the Munin version achieved a speedup of 7.6 on 16 processors. The cause for this dramatic difference in performance is Munin's ability to support multiple concurrent writers efficiently to a shared page of data. The message-passing version of FFT performed slightly better (speedup of 8.8 on 16 processors) than the Munin version.

The conventional DSM implementation takes over 300,000 faults, requires 1.35GB of data to be shipped and 1.65 million messages to be transmitted, and cumulatively spends over 25,000 seconds waiting for requests to be satisfied. Although not devoid of overhead, the Munin version requires orders of magnitude less communication. It only takes 2168 faults and reloads a total of 12MB of data. The primary source of overhead for the Munin program comes from sending out the updates during the data exchange phase after the first $\log_2 N - \log_2 P$ phases. At the beginning of the update phase, every processor is caching every page of shared data. This causes each processor to attempt to send updates for every page to every other processor, which adds two seconds of synchronization overhead. Munin's update timeout mechanism keeps the processors from actually shipping most of the data to every node, resulting in the Munin version shipping only slightly more data than the message-passing version.

6.6 Gaussian Elimination with Partial Pivoting

Program Description. Gaussian Elimination (GAUSS) decomposes a square matrix into upper and lower triangular submatrices by repeatedly eliminating the elements of the matrix under the diagonal, one column at a time. The basic algorithm for an $N \times N$ matrix is shown in Figure 13. For each iteration of the i loop, the algorithm subtracts the appropriate multiple of the ith row of the matrix from the rows below it, so that the elements below the diagonal in the ith column are zeroed. Partial pivoting improves the numerical stability of the basic algorithm by interchanging the ith row with the row in the range $[i + 1 \dots N - 1]$ containing the largest (in absolute value) element of the ith column. Algorithmically, this involves inserting a phase between the i and j loops that searches the ith column for the pivot element, and swapping that row and the ith row.

```
for i := 1 to N do
    for j := i+1 to N do
        for k = N+1 downto i do
            a[j][k] := a[j][k] - a[i][k]*a[j][i]/a[i][i];
```

Fig. 13. Basic (without pivoting) Gaussian elimination algorithm.

We decomposed the computation by column so that the pivoting phase, which can be a synchronization bottleneck, can be performed on a single processor. Each thread gets roughly $\lfloor N/P \rfloor$ columns, striped across the matrix, and any extra columns are spread evenly across the worker threads. The computation itself involves N iterations, one per column, each iteration consisting of a pivoting phase and a computation phase.

The DSM versions are parallelized as follows. The shared data structures are the array on which the elimination is being performed, a vector into which the pivot row is copied, and an integer that contains the number of the pivot row—all of which are declared to be `write_shared` in the Munin version. Each iteration starts with a barrier. After the barrier falls, the thread responsible for the current column performs the necessary pivoting, sets a shared pivot row variable to indicate the row that needs to be pivoted with the current one, and copies the current column to a shared variable to be used by the other threads during the computation phase. A barrier is used to separate the pivoting and computation phases. After the barrier is passed, each thread performs the actual computation, which involves performing the local pivoting, followed by the elimination step shown in Figure 13.

The message-passing version works similarly, except that the barrier is replaced by messages from the slaves to the central master, and the pivot column and pivot row number are explicitly sent to the workers rather than faulted in asynchronously.

Analysis. The DSM versions of GAUSS require two barriers per iteration for synchronization. The Munin version achieves a speedup of 8.6, 71% of the message-passing version's speedup of 12.1, on 16 processors. This reduced performance occurs because the relatively small amount of work done per iteration, particularly during the latter stages of the algorithm when there are very few nonzero elements left upon which to operate, accentuates the overhead imposed by both the general-purpose barrier mechanism and the need to update shared data during synchronization. On average, each thread spends over 40 seconds waiting for barriers, which includes the time spent exchanging data.

The conventional DSM version of GAUSS achieves a speedup of 5.1 on 16 processors, 42% of the message-passing version. In addition to the synchronization issues noted in the Munin implementation, the conventional DSM implementation also suffers from frequent read misses caused by accesses to invalidated data. Whereas the Munin implementation experiences 90 read misses, the conventional DSM implementation experiences 6780. This is

caused by the use of an invalidation-based consistency protocol in the conventional DSM system. Because all the modifications are made to shared data that are being actively shared (and constantly used) on all 16 processors, the update-pruning advantage of an invalidation protocol is not relevant, whereas the increased number of read misses is a significant problem. Each thread stalls for an average of 50 seconds for read misses to be serviced. In addition, because the last thread to have its read miss satisfied must wait until 14 other threads have successfully acquired their data, the computations tend to complete at noticeably different times. This causes the average time spent waiting at barriers to increase from 40 to 50 seconds. These two phenomena explain the lower performance of the conventional DSM implementation.

The performance times reported for the Munin version of all applications, including GAUSS, were with the update timeout mechanism enabled. For GAUSS, disabling the update timeout mechanism results in a slight performance advantage (a speedup of 8.9, instead of 8.6, on 16 processors). This is because, in GAUSS, all the modified data are accessed every iteration; thus it is best to propagate the updates and not invalidate selectively. In this case, the 50-millisecond default update timeout time was too short to ensure that no updates were timed out. Enabling the timeout mechanism thus resulted in unnecessary invalidations and subsequent reloads.

7. EFFECT OF COMMUNICATION REDUCTION TECHNIQUES

In this section we try to isolate the effects on performance of each technique for reducing communication described in Section 2. This isolation is made somewhat difficult because of the synergistic effect on performance of using the techniques in conjunction with one another. In particular, write-shared protocols cannot be used in the absence of release consistency or some other mechanism to relax memory consistency. Therefore, first we compare Munin's buffered write-update implementation of release consistency to a pipelined write-invalidate implementation of release consistency. Then we compare the use of multiple protocols versus using a single-protocol write-shared. Finally, we determine the value of the update timeout mechanism in connection with the update protocol.

7.1 Buffered Update Versus Pipelined Invalidate Release Consistency

In Section 2.1.1 we described the motivation for using a buffered update protocol for implementing release consistency in software and the advantages of doing so over using a pipelined invalidate protocol. To evaluate the performance impact of this decision, we implemented a pipelined write-invalidate consistency protocol and compared it to the buffered update protocol that is in normal use in Munin. In the pipelined write-invalidate protocol, a write fault causes ownership to be transferred to the faulting processor. Then invalidations are sent out in separate messages. Multiple invalidations can be outstanding concurrently, but no synchronization operation is allowed to complete until all outstanding invalidations have been acknowledged. We compared the performance of this implementation of release consistency with

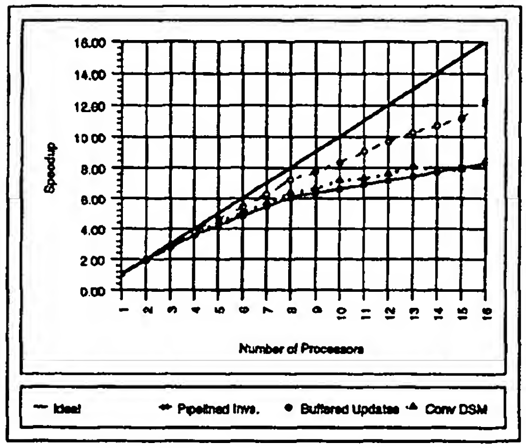

Fig. 14. Buffered write-update RC versus pipelined write-invalidate RC (DIFF).

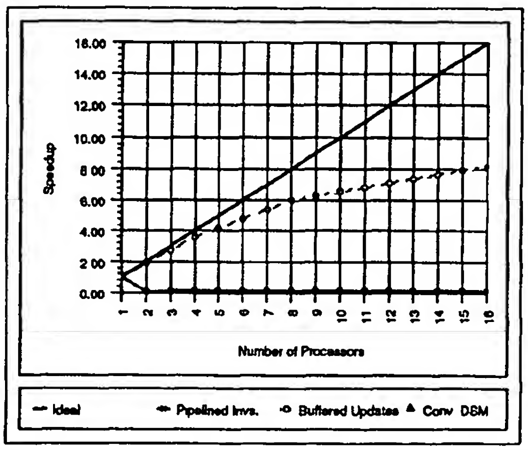

Fig. 15. Buffered write-update RC versus pipelined write-invalidate RC (FFT).

the Munin implementation using buffered update and with the conventional DSM system. For MULT, TSP-C, TSP-F, and GAUSS there is little difference between the pipelined write-invalidate and buffered write-update implementations of release consistency. For DIFF and QSORT, the buffered write-update scheme performs 30% better for 16 processors, whereas for FFT it performs orders of magnitude better. For the latter three applications, the pipelined write-invalidate protocol performs slightly better than a conventional write-invalidate protocol. Figures 14 and 15 depict these results for DIFF and FFT. The performance of QSORT is similar to that of DIFF.

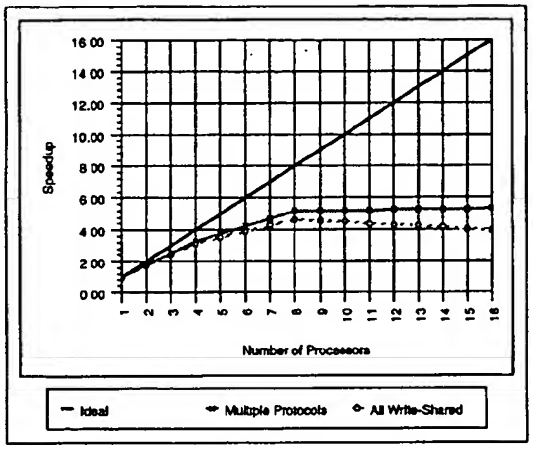

Fig 16. Multiprotocol versus all write-shared (TSP-F).

These results demonstrate that although the pipelined write-invalidate protocol offers some performance gain over a conventional sequentially consistent write-invalidate protocol in a software DSM system, a buffered write-update protocol outperforms both. Pipelining invalidations allows useful computation to be overlapped with invalidations, which reduces the cost of writes. However, it does not reduce the penalty associated with read misses, which are very expensive in a software DSM system. Furthermore, the pipelined-invalidate protocol suffers from false sharing, much in the same way that a conventional DSM system does. When read misses dominate, or when there is substantial false sharing, Munin's buffered update implementation is superior.

7.2 Multiple Consistency Protocols

To evaluate the importance of Munin's support for multiple consistency protocols, we compared the performance of two versions of Munin: a version in which multiple consistency protocols were used and a version that labeled all shared data as write-shared, thus employing Munin's most-versatile protocol. Figure 16 presents the results of this experiment for TSP-F; similar results were obtained for the other multiprotocol test programs (TSP-C and QSORT). For TSP-F, using multiple protocols leads to a 30% improvement in speedup for 16 processors. The reason is that the multiprotocol version of the program declares the task queue to be migratory, resulting in the advantages described in Section 2.2. Although a 30% improvement in performance is modest, the cost associated with implementing multiple protocols in a software DSM system is essentially zero.

7.3 Update Timeout Mechanism

To test the value of the timeout mechanism in connection with the update protocol, we compared the performance of versions with and without the

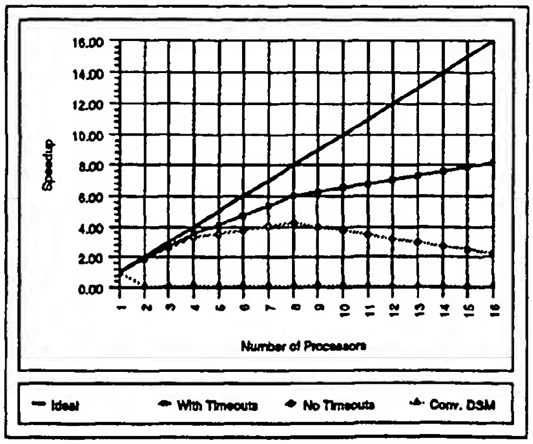

Fig. 17. Effect of update timeout mechanism on FFT.

timeout enabled. For MULT, DIFF, and TSP-C there is no difference. For TSP-F and QSORT, the version with the timeout enabled is 10% and 15% faster for 16 processors, respectively. The difference is the largest for FFT. Speedup with 16 processors drops from 8.2 to 3.6 when the timeout is disabled (see Figure 17). Finally, for GAUSS, the timeout causes a 5% dropoff in performance for 16 processors.

In terms of the underlying DSM operation, without the timeout mechanism the 16-processor FFT sends 120,000 messages and 109MB of data, whereas, with the timeout mechanism enabled, the 16-processor FFT sends only 48,000 messages and 78MB of data. The reason that the amount of data shipped does not drop as dramatically as the number of messages is that, after a page of data has been speculatively invalidated, future accesses require an 8KB page to be transferred rather than just a *diff*.

The other two programs in which each processor's working set changes dynamically over the course of the program execution, TSP and QSORT, are also aided by the use of the timeout mechanism. For TSP, each page of the shared tour array tends to be used by many different processors over time, but each processor only uses it for a very short period of time, and only a few processors use a particular page at a time. Without the timeout mechanism, eventually almost every processor receives updates for almost every page. The shared sort array in QSORT exhibits a similar phenomenon.

With GAUSS, all the modified data are accessed every iteration. The slight dropoff in performance for GAUSS is caused by the fact that the default update timeout time of 50 milliseconds is too short to ensure that no valid updates are timed out.

8. FUNCTION SHIPPING

For TSP-F and QSORT, the two programs that use the task queue model of parallelism and that have a significant amount of sharing, the Munin 16-

Table V. Lock Waiting Times for TSP-F and QSORT

Program	Average Lock Waiting Time per Processor (seconds)	Execution Time per Processor (seconds)
Munin TSP-F	19	32
Conventional TSP-F	22	45
Munin QSORT	53	135
Conventional QSORT	13	310

processor versions achieved speedups of only 6.0 and 8.9, respectively, compared to 8.9 and 13.4 for the message-passing versions. The conventional DSM versions performed even worse, achieving speedups of 4.7 and 4.1, respectively. As shown in Table V, the major source of overhead for these DSM versions (with the exception of the conventional version of QSORT) is the amount of time spent waiting on the lock protecting the work queues. For the conventional version of QSORT, false sharing within the array being sorted is the dominant source of overhead.

These lock waiting times are large because the DSM versions must ship the work queue, a sizable data structure, to the acquiring thread before that thread can perform any operation on the work queue. In comparison, the actual time spent performing operations on the work queue is very small. The message-passing versions do not suffer from this phenomenon, since the work queue is kept at the root node and since worker threads perform remote procedure calls (RPCs), containing only a small amount of data, to the root node in order to operate on the queue.

In order to evaluate the feasibility and potential value of using a mixed data-shipping and function-shipping mechanism in a DSM system, we modified the DSM versions of TSP-F and QSORT such that the task queue remains attached to the root node, and all access to the task queue by other nodes is performed using RPC. These modifications were done in an ad hoc manner, but research is ongoing to extend Munin to support both DSM and function shipping in an integrated fashion. The results of function-shipping access to the task queue for the TSP-F and QSORT are shown in Figures 18 and 19. These figures show the speedups achieved by Munin and conventional DSM both with and without function shipping for the task queue.

For TSP-F, function shipping causes both DSM versions to perform almost as well as the message-passing version (on 16 processors, a speedup of 9.1 for conventional DSM, 9.8 for Munin, and 10.6 for message passing). In contrast, without function shipping, Munin achieves a speedup of only 6.0, and the conventional DSM a speedup of only 4.7. For the Munin version without function shipping, communication is substantially more (9229 messages and 4989KB of data) than the Munin version with function shipping (3630 messages and 888KB of data). Perhaps more importantly, the reduced communication of the function-shipping version nearly eliminates the time that threads are idle waiting for access to the task queue.

For QSORT, improvements are similar to those in TSP-F for the Munin version, but no improvement is achieved for the conventional DSM version.

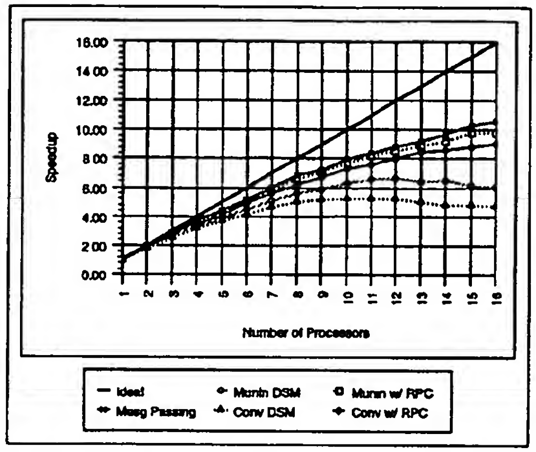

Fig. 18. Effect of function shipping on fine-grained TSP.

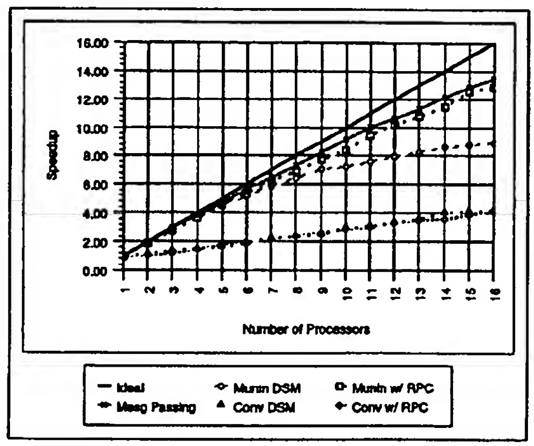

Fig. 19. Effect of function shipping on Quicksort.

The addition of function shipping for the task queue raises the 16-processor speedup for Munin from 8.9 to 12.9, compared to 13.4 for the message-passing version. The conventional DSM version, both with and without function shipping for the task queue, achieved only a speedup of 4.1. As explained in Section 6, false sharing is the primary obstacle to good performance for the conventional version. Although the average time waiting for locks is reduced from 13 seconds to below 1 second, the average time a process waits for fresh copies of data increases from 145 to 176 seconds, so the addition of function shipping has no beneficial effects.

These experiments show that the addition of function shipping for accessing some shared data can improve the performance of some programs signifi-

cantly. In addition, the QSORT experiment further illustrates the value of Munin's write-shared protocol for dealing with false sharing.

9. RELATED WORK

This section compares our work with a number of existing software and hardware DSM systems, focusing on the mechanisms used by these other systems to reduce the amount of communication necessary to provide shared memory. We limit our discussion to those systems that are most related to the work presented in this article.

9.1 Software DSMs

Ivy was the first software DSM system [Li and Hudak 1989]. It uses a single-writer, write-invalidate protocol for all data, with virtual memory pages as the units of consistency. This protocol is used as the baseline conventional protocol in our experiments. The large size of the consistency unit and the single-writer protocol makes the system prone to large amounts of communication due to false sharing. It is up to the programmer or the compiler to lay out the program data structures in the shared address space such that false sharing is reduced. The directory management scheme in our implementation is largely borrowed from Ivy's dynamic distributed manager scheme.

Both Clouds [Dasgupta et al. 1990] and Mirage [Fleisch and Popek 1989] allow part of shared memory to be locked down at a particular processor. In Clouds, the programmer can request that a segment of shared memory be locked on a processor. In Mirage, a page remains at a processor for a certain Δ time window after it is modified by that processor. In both cases, the goal is to avoid extensive communication due to false sharing. The combination of software release consistency and write-shared protocols addresses the adverse effects of false sharing without introducing the delays caused by locking parts of shared memory to a processor.

Mether [Minich and Farber 1989] supports a number of special shared-memory segments in fixed locations in the virtual address space of each machine in the system. In an attempt to support efficient memory-based spinlocks, Mether supports several different shared-memory segments, each with different protocol characteristics. Two segments are for small objects (up to 32 bytes), and two are for large objects (up to 8192 bytes). One of each pair is "demand driven," which means that the memory is shipped when it is read, as in a conventional DSM. The other is "data driven," which means that it is shipped when it is written. A thread that attempts to read the data will block until the next thread writes it. This latter form of data can support spinlocks and message passing fairly effectively. Our support for multiple protocols is more general, without added cost, and Munin's separate synchronization package removes the need to support data-driven memory.

Lazy release consistency, as used in TreadMarks [Keleher et al. 1994], is an algorithm for implementing release consistency different from the one presented in this article. Instead of updating every cached copy of a data item

whenever the modifying thread performs a release operation, only the cached copies on the processor that next acquires the released lock are updated. Lazy release consistency reduces the number of messages required to maintain consistency, but the implementation is more expensive in terms of protocol and memory overhead [Keleher et al. 1992].

A variety of systems have sought to present an object-oriented interface to shared memory. We describe the Orca [Bal et al. 1992] as an example of this approach. In general, the object-oriented nature allows the compiler and the runtime system to carry out a number of powerful optimizations, but the programs have to be written in the particular object model supported.

The Orca language requires that (a) all access to objects is through well-defined per-object operations, (b) only one operation on an object can be performed at a time, and (c) there are no global variables or pointers. This programming model allows the compiler to detect all accesses to an object directly without the use of page faults. Programmers must, however, structure their programs so that objects are accessed in a way that does not limit performance. For example, an Orca implementation of DIFF requires that the edge elements be specified as shared buffers—the entire array should not be declared as a single object. However, once a program has been structured appropriately, Orca can transparently choose whether to replicate an object or force all accesses to be made via RPCs to a master node. If it chooses to replicate an object, it can support both invalidate and update consistency protocols. It remains to be seen how well Orca's optimizations can be integrated into a less-restrictive language. On an orthogonal issue, Orca's consistency management uses an efficient, reliable, ordered broadcast protocol. For reasons of scalability, Munin does not rely on broadcast, although support for efficient multicast could improve the performance of some aspects of Munin.

Midway [Bershad et al. 1993] proposes a DSM system with *entry consistency*, a memory consistency model weaker than release consistency. The goal of Midway is to minimize communication costs by aggressively exploiting the relationship between shared variables and the synchronization objects that protect them. Entry consistency only guarantees the consistency of a data item when the lock associated with it is acquired. To exploit the power of entry consistency, the programmer must associate each individual unit of shared data with a single lock. For some programs, making this association is easy. However, for programs that use nested data structures or arrays, it is not clear if making a one-to-one association is feasible without forcing programmers to completely rewrite their programs. For example, the programmer of an entry-consistent DIFF program would have to hand-decompose the shared array to exploit the power of entry consistency. The designers of Midway recognized this problem and proposed to give programmers the ability to increase and decrease the strength of the consistency model supported. Thus, programs for which the data synchronization association required by entry consistency is convenient can exploit its flexibility, whereas programs for which this association is inconvenient can use either release consistency (when adequate synchronization is performed) or sequential consistency. Unlike Munin, Midway exploits the power of a sophisticated com-

piler. The Midway compiler inserts code around data accesses so that the Midway runtime system can determine whether a particular shared variable is present before it is accessed. Thus Midway is able to detect access violations without taking page faults, which eliminates the time spent handling interrupts.

9.2 Hardware DSMs

Several designs for distributed shared-memory hardware systems have been published recently, of which DASH [Gharachorloo et al. 1990], GalacticaNet [Wilson and La Rowe 1992], and APRIL [Agarwal et al. 1990] are representative.

We have adopted from the DASH project [Gharachorloo et al. 1990] the concept of release consistency. The differences between DASH's implementation of release consistency and Munin's implementation of release consistency were explained in detail in Section 2.1. DASH uses a write-invalidate protocol for all consistency maintenance. We use instead the flexibility of its software implementation to attack the problem of read misses also by using update protocols and migration when appropriate. The GalacticaNet system [Wilson and La Rowe 1992] also demonstrates that support for an update-based protocol that exploits the flexibility of a relaxed consistency protocol can improve performance by reducing the number of read misses and attendant processor stalls. The GalacticaNet design includes a provision to timeout updates to stale data, which is shown to have a significant effect on performance when there is a large number of processors.

The APRIL machine addresses the problem of high latencies in distributed shared-memory multiprocessors in a different way [Agarwal et al. 1990]. APRIL provides sequential consistency, but relies on extremely fast processor switching to overlap memory latency with computation. For APRIL to be successful at reducing the impact of read misses, there must be several threads ready to run on each processor. Because APRIL performs many low-level consistency operations in very fast trap-handling software, it would be possible to adopt several of our techniques to their hardware cache consistency mechanism.

10. CONCLUSIONS AND DIRECTIONS FOR FURTHER WORK

Distributed shared-memory (DSM) software systems provide a shared-memory abstraction on hardware with physically distributed memory. This approach is appealing because it combines the desirable features of distributed- and shared-memory machines: distributed-memory machines are easier to build, but shared memory provides a more-convenient programming model. It has, however, proven to be difficult to achieve performance on DSM systems that is comparable to what can be achieved with hand-coded message-passing programs. In particular, conventional DSM implementations have suffered from excessive amounts of communication engendered by sequential consistency and false sharing.

In this article we have presented and evaluated a number of techniques to reduce the amount of communication necessary to maintain consistency. In particular, we replaced sequential consistency by release consistency as our choice of consistency model. We developed a buffered, update-based implementation of release consistency, suitable for software systems. The update protocol has a timeout feature, preventing large numbers of unnecessary updates to copies of pages that are no longer in use. Furthermore, we allow the use of multiple protocols to maintain consistency. Of particular interest among these protocols is the write-shared protocol that allows several processes to write to a page concurrently, with the individual modifications merged at a later point according to the requirements of release consistency.

We have implemented these techniques in the Munin DSM system. The resulting system runs on a network of workstations and provides an interface that is very close to a conventional shared-memory programming system. For programs that are free of data races, release-consistent memory produces the same results as sequentially consistent memory. All synchronization operations must be performed through system-supplied primitives, and shared variables may optionally be annotated with the desired consistency protocol. For the applications that we have studied, these requirements proved to be a very minor burden.

The use of these techniques has substantially broadened the class of applications for which DSM on a network of workstations is a viable vehicle for parallel programming. For very coarse grained applications conventional DSM performs satisfactorily. However, as the granularity of parallelism decreases, conventional DSM performance falls behind, whereas Munin's performance continues to track that of hand-coded message passing. The addition of a function-shipping ability improves the performance of DSM further.

Hardware technology has improved dramatically since the experiments reported here were performed, and there are no signs that the current rate of performance improvement will abate soon. In particular, both processor and network speeds have improved by a factor of 15 to 20 in the past four years. Interprocessor communication is still a high-latency operation, but there are indications that latencies can be improved by an order of magnitude through careful protocol implementation [Thekkath and Levy 1993]. At the same time, DRAM latencies are improving very slowly, so some form of cache will be present on essentially all future high-performance platforms. Finally, hardware DSM systems are becoming more common. An important issue to address is the applicability of the techniques introduced in this article to future DSM systems, both hardware and software.

We believe that there are two basic requirements that DSM systems, hardware or software, must satisfy to provide acceptably high performance. Both the *latency* and the *frequency* of processor-stalling DSM operations (e.g., cache misses or synchronization events) must be kept low. It appears that despite improvements in networking and operating system designs, the latency of remote operations will slowly increase compared to processor cycle times. However, because memory speeds are not increasing very rapidly, the

ratio of remote memory access to local memory access (not satisfied by the cache) will decrease. This observation would seem to indicate that a simple implementation of DSM that ships entire pages (or cache lines) on demand and uses invalidation to maintain consistency would suffice as processor and network technology improves. We believe that this will not be the case because of our second requirement for efficient DSM: a low *frequency* of processor-stalling DSM operations. As processor cycle times continue to decrease dramatically, it is becoming increasingly important to avoid stalling the processor. As described in Section 7.1, using a conventional invalidation-based consistency protocol can increase the number of high-latency read misses dramatically. Also, as the size of memories and caches increases, page and cache line sizes are also increasing, which indicates that false sharing will become an increasingly important problem. These observations indicate that some form of update protocol that supports multiple concurrent writers, such as Munin's write-shared protocol, will be useful in future DSM systems.

Our current DSM work focuses on techniques required to implement DSM on current high-performance platforms, with faster processors and networks than the ones used for the experiments in this article. In particular, we are studying a more-aggressive implementation of release consistency—lazy release consistency—and compiler techniques to optimize performance further. We are also studying the value of the techniques described here in the context of hardware-supported distributed shared-memory multiprocessors.

REFERENCES

AGARWAL, A., LIM, B.-H., KRANZ, D., AND KUBIATOWICZ, J. 1990. APRIL: A processor architecture for multiprocessing In *Proceedings of the 17th Annual International Symposium on Computer Architecture* (Seattle, Wash., May). IEEE Computer Society Press, Los Alamitos, Calif., 104–114.

ARCHIBALD, J. AND BAER, J.-L. 1986. Cache coherence protocols: Evaluation using a multiprocessor simulation model. *ACM Trans. Comput. Syst. 4*, 4 (Nov.), 273–298.

BAL, H. E., KAASHOECK, M. F., AND TANENBAUM, A. S. 1992. Orca: A language for parallel programming of distributed systems. *IEEE Trans. Softw. Eng. 18*, 3 (Mar.), 190–205.

BENNETT, J. K., CARTER, J. B., AND ZWAENEPOEL, W. 1990. Adaptive software cache management for distributed shared memory architectures. In *Proceedings of the 17th Annual International Symposium on Computer Architecture* IEEE Computer Society Press, Los Alamitos, Calif., 125–134.

BERSHAD, B. N., ZEKAUSKAS, M. J., AND SAWDON, W. A. 1993. The Midway distributed shared memory system. In *COMPCON '93*. IEEE Computer Society Press. Los Alamitos, Calif., 528–537.

CARTER, J. B. 1993. Efficient distributed shared memory based on multi-protocol release consistency. Ph D. thesis, Rice Univ., Houston, Tex. Aug.

CARTER, J. B., BENNETT, J. K., AND ZWAENEPOEL, W. 1991. Implementation and performance of Munin. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles* (Oct.) ACM, New York, 152–164.

CHASE, J. S., AMADOR, G. F., LAZOWSKA, E. D., LEVY, H. M., AND LITTLEFIELD, R. J. 1989. The Amber system: Parallel programming on a network of multiprocessors. In *Proceedings of the 12th ACM Symposium on Operating Systems Principles* (Dec.). ACM. New York. 147–158.

CHERITON, D. R. 1988. The V distributed system. *Commun. ACM 31*, 3 (Mar.), 314–333.

DASGUPTA, P., CHEN, R. C., MENON, S., PEARSON, M., ANANTHANARAYANAN, R., RAMACHANDRAN, U., AHAMAD, M., LEBLANC, R., JR., APPLEBE, W., BERNABEU-AUBAN, J. M., HUTTO, P. W., KHALIDI,

M. Y. A., AND WILEKNLOH, C. J. 1990. The design and implementation of the Clouds distributed operating system. *Comput. Syst. J. 3*, (Winter).

EGGERS, S. J. AND KATZ, R. H. 1988. A characterization of sharing in parallel programs and its application to coherency protocol evaluation. In *Proceedings of the 15th Annual International Symposium on Computer Architecture* (May). IEEE Computer Society Press, Los Alamitos, Calif., 373–383.

FLEISCH, B. AND POPEK, G. 1989. Mirage: A coherent distributed shared memory design. In *Proceedings of the 12th ACM Symposium on Operating Systems Principles* (Dec.). ACM, New York, 211–223.

GHARACHORLOO, K., GUPTA, A., AND HENNESSY, J. 1991. Performance evaluations of memory consistency models for shared-memory multiprocessors. In *Proceedings of the 4th Symposium on Architectural Support for Programming Languages and Operating Systems* (Apr.). ACM, New York.

GHARACHORLOO, K., LENOSKI, D., LAUDON, J., GIBBONS, P., GUPTA, A., AND HENNESSY, J. 1990. Memory consistency and event ordering in scalable shared-memory multiprocessors. In *Proceedings of the 17th Annual International Symposium on Computer Architecture* (Seattle, Wash., May). IEEE Computer Society Press, Los Alamitos, Calif., 15–26.

JUL, E., LEVY, H., HUTCHINSON, N., AND BLACK, A. 1988. Fine-grained mobility in the Emerald system. *ACM Trans. Comput. Syst. 6*, 1 (Feb.), 109–133.

KARLIN, A. R., MANASSE, M. S., RUDOLPH, L., AND SLEATOR, D. D. 1986. Competitive snoopy caching. In *Proceedings of the 16th Annual IEEE Symposium on the Foundations of Computer Science*. IEEE, New York, 244–254.

KELEHER, P., COX, A. L., AND ZWAENEPOEL, W. 1992. Lazy consistency for software distributed shared memory. In *Proceedings of the 19th Annual International Symposium on Computer Architecture* (May). IEEE Computer Society Press, Los Alamitos, Calif., 13–21.

KELEHER, P., DWARKADAS, S., COX, A., AND ZWAENEPOEL, W. 1994. Treadmarks: Distributed shared memory on standard workstations and operating systems. In *Proceedings of the 1994 Winter USENIX Conference* (Jan.). USENIX Assoc., Berkeley, Calif., 115–131.

LAMPORT, L. 1979. How to make a multiprocessor computer that correctly executes multiprocess programs. *IEEE Trans. Comput. C-28*, 9 (Sept.), 690–691.

LENOSKI, D., LAUDON, J., GHARACHORLOO, K., GUPTA, A., AND HENNESSY, J. 1990. The directory-based cache coherence protocol for the DASH multiprocessor. In *Proceedings of the 17th Annual International Symposium on Computer Architecture* (Seattle, Wash.. May). IEEE Computer Society Press, Los Alamitos, Calif., 148–159.

LI, K. AND HUDAK, P. 1989. Memory coherence in shared virtual memory systems. *ACM Trans. Comput. Syst. 7*, 4 (Nov.), 321–359.

MINNICH, R. G. AND FARBER, D. J. 1989. The Mether system: A distributed shared memory for SunOS 4.0. In *Proceedings of the Summer USENIX Conference* (June). USENIX Assoc., Berkeley, Calif., 51–60.

THEKKATH, A. C. AND LEVY, H. 1993. Limits to low-latency communications on high-speed networks. *ACM Trans. Comput. Syst. 11*, 2 (May), 179–203.

WEBER, W.-D. AND GUPTA, A. 1989. Analysis of cache invalidation patterns in multiprocessors. In *Proceedings of the 3rd Symposium on Architectural Support for Programming Languages and Operating Systems* (Apr.). ACM, New York, 243–256.

WILSON, A. AND LAROWE, R. 1992. Hiding shared memory reference latency on the GalacticaNet distributed shared memory architecture. *J. Parallel Distrib. Comput. 15*, 4 (Aug.), 351–367.

Received May 1993; revised June 1994; accepted January 1995

```
File   8:Ei Compendex(R) 1970-2005/Dec W4
         (c) 2005 Elsevier Eng.  Info. Inc.
File  35:Dissertation Abs Online 1861-2004/Dec
         (c) 2004 ProQuest Info&Learning
File  65:Inside Conferences 1993-2005/Jan W1
         (c) 2005 BLDSC all rts. reserv.
File   2:INSPEC 1969-2004/Dec W2
         (c) 2004 Institution of Electrical Engineers
File  94:JICST-EPlus 1985-2004/Nov W4
         (c)2004 Japan Science and Tech Corp(JST)
File 483:Newspaper Abs Daily 1986-2005/Jan 05
         (c) 2005 ProQuest Info&Learning
File   6:NTIS 1964-2004/Dec W4
         (c) 2004 NTIS, Intl Cpyrght All Rights Res
File 144:Pascal 1973-2004/Dec W1
         (c) 2004 INIST/CNRS
File 434:SciSearch(R) Cited Ref Sci 1974-1989/Dec
         (c) 1998 Inst for Sci Info
File  34:SciSearch(R) Cited Ref Sci 1990-2005/Jan W1
         (c) 2005 Inst for Sci Info
File  99:Wilson Appl. Sci & Tech Abs 1983-2004/Nov
         (c) 2004 The HW Wilson Co.
File 583:Gale Group Globalbase(TM) 1986-2002/Dec 13
         (c) 2002 The Gale Group
File 266:FEDRIP 2004/Sep
         Comp & dist by NTIS, Intl Copyright All Rights Res
File  95:TEME-Technology & Management 1989-2004/Jun W1
         (c) 2004 FIZ TECHNIK
File 438:Library Lit. & Info. Science 1984-2004/Oct
         (c) 2004 The HW Wilson Co
File  62:SPIN(R) 1975-2004/Oct W4
         (c) 2004 American Institute of Physics
File 239:Mathsci 1940-2004/Feb
         (c) 2004 American Mathematical Society

Set     Items   Description
S1     338817   CACH??? OR BUFFER???
S2     471076   BIN OR BINS OR BOX OR BOXES OR CONTAINER? ? OR BUCKET? ?
S3      52973   (PAST OR OLD??? OR PRIOR? OR PREVIOUS? OR PRECED??? OR EAR-
             LY OR EARLIE?? OR FORMER OR AGED OR AGING OR HISTOR???)(7N)(V-
             ERSION? ? OR EDITION? ?)
S4      86563   (DIFFERENT OR VARIOUS OR SEPARATE OR ALTERNAT??? OR SECOND
             OR 2ND OR TWO OR MULTIPLE OR MULTIPLICITY OR SEVERAL OR MANY -
             OR PLURAL? OR ASSORT???? OR ADDITIONAL)(7N)(VERSION? ? OR EDI-
             TION? ?)
S5        262   S2(10N)S3:S4
S6          1   S1 AND S5
S7       1423   (VERSION? ? OR EDITION? ?)(10N)S2
S8          9   S1 AND S7
S9         17   S5 AND (DATABASE? ? OR DATA()BASE? ? OR DBMS OR RDBMS OR R-
             EPOSITOR??? OR LIBRAR??? OR ARCHIVE? ? OR (INFORMATION OR DAT-
             A)()MANAG??? OR DATE()(MART? ? OR WAREHOUSE? ?) OR DATAMART? ?
              OR CLUSTER???)
S10        26   S6 OR S8 OR S9
S11        19   RD (unique items)
```

11/5/1 (Item 1 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

06306174 E.I. No: EIP03097373381
Title: Visualizing impacts of database **schema changes - A controlled experiment**
Author: Karahasanovic, Amela; Sjoberg, Dag I.K.
Corporate Source: Industrial Systems Development Group Department of Informatics University of Oslo, Oslo, Norway
Conference Title: Proceedings: IEEE Symposia on Human-Centric Computing Languages and Environments
Conference Location: Stresa, Italy Conference Date: 20010905-20010907
Sponsor: IEEE Computer Society
E.I. Conference No.: 60561
Source: 2001 IEEE Symposium on Human-Centric Computing 2001.
Publication Year: 2001
ISBN: 0780371984
Language: English
Document Type: CA; (Conference Article) Treatment: T; (Theoretical); X; (Experimental)
Journal Announcement: 0303W1
Abstract: Research in schema evolution has been driven by the need for more effective software development and maintenance. Finding impacts of schema changes on the applications and presenting them in an appropriate way are particularly challenging. We have developed a tool that finds impacts of schema changes on applications in object-oriented systems. This tool displays components (packages, classes, interfaces, methods and fields) of a **database** application system as a graph. Components potentially affected by a change are indicated by changing the shape of the **boxes** representing those components. **Two versions** of the tool are available. One **version** identifies affected parts of applications at the granularity of packages, classes, and interfaces, whereas the other version identifies affected parts at the finer granularity of fields and methods. This paper presents the design and results of a controlled student experiment testing these two granularity levels with respect to productivity and user satisfaction. There are indications that identifying impacts at the finer granularity can reduce the time needed to conduct schema changes and reduce the number of errors. Our results also show that the subjects of the experiment appreciated the idea of visualizing the impacts of schema changes. 45 Refs.
Descriptors: ***Databas** e systems; Software engineering; Java programming language; Interfaces (computer); Visualization
Identifiers: Schema evolution management tool (SEMT)
Classification Codes:
723.1.1 (Computer Programming Languages)
723.3 (Database Systems); 723.1 (Computer Programming); 722.2 (Computer Peripheral Equipment)
723 (Computer Software, Data Handling & Applications); 722 (Computer Hardware)
72 (COMPUTERS & DATA PROCESSING)

11/5/2 (Item 2 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

05402433 E.I. No: E2099104857022
Title: Comparison of meta-heuristic algorithms for clustering **rectangles**
Author: Burke, Edmund; Kendall, Graham
Corporate Source: Univ of Nottingham, Nottingham, Engl
Conference Title: Proceedings of the 1998 24th International Conference on Computers and Industrial Engineering
Conference Location: Middlesex, UK Conference Date: 19980902-19980904
E.I. Conference No.: 55481
Source: Computers and Industrial Engineering v 37 n 1 1999. p 383-386
Publication Year: 1999

CODEN: CINDDL   ISSN: 0360-8352
Language: English
Document Type: JA; (Journal Article)   Treatment: T; (Theoretical)
Journal Announcement: 9912W1
Abstract: In this paper we consider a simplified **version** of the stock cutting (**two** -dimensional **bin** packing) problem. We compare three meta-heuristic algorithms (genetic algorithm (GA), tabu search (TS) and simulated annealing (SA)) when applied to this problem. The results show that tabu search and simulated annealing produce good quality results. This is not the case with the genetic algorithm. The problem, and its representation, is fully described along with key test results. (Author abstract) 4 Refs.
Descriptors: *Optimization; Genetic algorithms; Simulated annealing; Heuristic methods; Asymptotic stability; Probability
Identifiers: Meta heuristic algorithms; Stock cutting; Tabu search; Two dimensional bin packing
Classification Codes:
921.5 (Optimization Techniques); 723.1 (Computer Programming); 922.1 (Probability Theory)
921 (Applied Mathematics); 723 (Computer Software); 922 (Statistical Methods)
92 (ENGINEERING MATHEMATICS); 72 (COMPUTERS & DATA PROCESSING)

11/5/3 (Item 3 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

04201665 E.I. No: EIP95072765273
Title: Improved dual leaky bucket policing algorithm for ATM networks
Author: Mannem, V.; Sankar, R.
Corporate Source: Univ of South Florida, Tampa, FL, USA
Conference Title: Proceedings of the IEEE Southeastcon '95 Conference
Conference Location: Raleigh, NC, USA Conference Date: 19950326-19950329
E.I. Conference No.: 43203
Source: Conference Proceedings - IEEE SOUTHEASTCON 1995.,95CH35793. p 6-11
Publication Year: 1995
CODEN: CPISDM
Language: English
Document Type: CA; (Conference Article)   Treatment: A; (Applications); T ; (Theoretical)
Journal Announcement: 9509W1
Abstract: In high speed communication, traffic management plays a significant role to control the use of network resources to prevent bottlenecks. In particular, when the network resources are allotted to more sources than the network can support, the network performance degrades for users (for ex. **buffer** overflows and cell delay occurs). So, the network resources should be allotted in such a way that the network can operate at acceptable levels (maintain the QoS for all the existing connections) at times when the offered load to the network exceeds its capacity. So, a policing function is setup at the User Network Interface (UNI) for preventive congestion control. Lot of algorithms are proposed for the UNI. Only a few algorithms are concerned with the cell delay variation (CDV). In this paper, an improved **version** of the leaky **bucket** algorithm is presented to monitor traffic. This algorithm also checks those cells which are crossing the policing rate due to the effect of CDV and compensates them by sending them into the network without discarding. Simulation results are presented to demonstrate the superior performance of this traffic control algorithm. The algorithm also makes use of the feedback control of the network status which in turn improves the throughput. (Author abstract) 6 Refs.
Descriptors: *Asynchronous transfer mode; Algorithms; User interfaces; Computer simulation; Feedback control; Bandwidth; Packet networks; Data acquisition; Telecommunication traffic; Performance
Identifiers: User network interfaces; Cell delay variations; Leaky bucket algorithm

Classification Codes:
716.1 (Information & Communication Theory); 723.1 (Computer Programming); 722.2 (Computer Peripheral Equipment); 723.5 (Computer Applications); 731.1 (Control Systems); 703.1 (Electric Networks)
716 (Radar, Radio & TV Electronic Equipment); 723 (Computer Software); 722 (Computer Hardware); 731 (Automatic Control Principles); 703 (Electric Circuits)
71 (ELECTRONICS & COMMUNICATIONS); 72 (COMPUTERS & DATA PROCESSING); 73 (CONTROL ENGINEERING); 70 (ELECTRICAL ENGINEERING)

11/5/4 (Item 4 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

03028708 E.I. Monthly No: EI9103024819
Title: Implementation of Pro Test. A Prolog debugger for a refined box model.
Author: Schleiermacher, A.; Winkler, J. F. H.
Corporate Source: Siemens AG, Munich, West Ger
Source: Software - Practice and Experience v 20 n 10 Oct 1990 p 985-1006
Publication Year: 1990
CODEN: SPEXBL ISSN: 0038-0644
Language: English
Document Type: JA; (Journal Article) Treatment: T; (Theoretical); A; (Applications)
Journal Announcement: 9103
Abstract: We describe some aspects of the implementation of a Prolog debugger for a refined box model in which attempted unifications can also be observed. Our implementation of the ProTest debugger is based on a meta-interpreter for Prolog. We start with an existing meta-interpreter for Byrd's box model (four-port debugger) and we transform it into one for the refined **box** model (ten-port debugger). To explain the transformation we show **several versions** of the meta-interpreter. In these **versions** we use the technique of changing the **database** to implement the cut, but another possibility is also explained briefly. A simple notation for typing is used to make Prolog programs more readable. In an appendix we give a listing of a simple prototype of the extended meta-interpreter. (Author abstract) 32 Refs.
Descriptors: *COMPUTER PROGRAMMING--*Program Debugging; COMPUTER PROGRAMMING LANGUAGES--Prolog; COMPUTER OPERATING SYSTEMS--Program Interpreters
Identifiers: REFINED BOX MODEL; META INTERPRETERS
Classification Codes:
723 (Computer Software); 722 (Computer Hardware)
72 (COMPUTERS & DATA PROCESSING)

11/5/5 (Item 5 from file: 8)
DIALOG(R)File 8:Ei Compendex(R)

02287038 E.I. Monthly No: EI8707067496
Title: PARTIAL EXPANSIONS FOR FILE ORGANIZATIONS WITH AN INDEX.
Author: Lomet, David B.
Corporate Source: Wang Inst of Graduate Studies, Tyngboro, MA, USA
Source: ACM Transactions on Database Systems v 12 n 1 Mar 1987 p 65-84
Publication Year: 1987
CODEN: ATDSD3 ISSN: 0362-5915
Language: ENGLISH
Document Type: JA; (Journal Article) Treatment: E; (Economic/Cost Data/Market Survey); T; (Theoretical)
Journal Announcement: 8707
Abstract: A new way to increase file space in dynamically growing files is introduced in which substantial improvement in file utilization can be achieved. It makes use of partial expansions in which, instead of doubling the space associated with some part of the file, the space grows at a slower rate. Unlike **previous versions** of partial expansion in which the

number of **buckets** involved in file growth is increased by less than a factor of two, the new method expands file space by increasing bucket size via 'elastic buckets. ' This permits partial expansions to be used with a wide range of indexed files, including B-trees. The results of using partial expansions are analyzed, and the analysis confirmed by a simulation study. The analysis and simulation demonstrate that the file utilization gains are substantial and that fears of excessive insertion cost resulting from more frequent file growth are unfounded. (Author abstract) 13 refs.

Descriptors: *DATA PROCESSING--*File Organization; **DATABASE** SYSTEMS
Identifiers: PARTIAL EXPANSIONS; FILE UTILIZATION; INSERTION COST
Classification Codes:
723 (Computer Software)
72 (COMPUTERS & DATA PROCESSING)

11/5/6 (Item 1 from file: 35)
DIALOG(R)File 35:Dissertation Abs Online

01298815 ORDER NO: NOT AVAILABLE FROM UNIVERSITY MICROFILMS INT'L.
PLAY AND THE GENESIS OF MIDDLE MANAGER AGENTS (COGNITION)
Author: MCGEE, KEVIN
Degree: PH.D.
Year: 1993
Corporate Source/Institution: MASSACHUSETTS INSTITUTE OF TECHNOLOGY (0753)
Supervisor: SEYMOUR PAPERT
Source: VOLUME 54/03-B OF DISSERTATION ABSTRACTS INTERNATIONAL. PAGE 1694.
Descriptors: PSYCHOLOGY, DEVELOPMENTAL; EDUCATION, PSYCHOLOGY; ARTIFICIAL INTELLIGENCE
Descriptor Codes: 0620; 0525; 0800

Play-like behaviors result in substantial changes to an individual's way of knowing, changes that are difficult to explain in such traditional terms as "problem solving," or reinforcement and reward.

Developmental psychologist Jean Piaget discovered that during the course of development individuals construct fundamentally new ways of understanding the world. In one of his famous clinical experiments, Piaget and his associates asked individuals of different ages what would happen to liquids as they were poured from one **container** to another. In one **version** of this experiment, three clear glasses, **two** the same size and one taller and thinner, are presented. Each of the two similar glasses are filled with equal amounts of liquid. When asked "is there one that has more?" most individuals say that they have the same amount. However, if, before their very eyes, the liquid is poured from one of the shorter glasses into the tall, thin glass and they are then asked "is there one that has more?" individuals before a certain stage give a surprising answer. They say that the tall glass has more.

Professors Minsky and Papert developed a Society of Mind model of this phenomenon which says that individuals have cognitive agents for, among other things, measuring liquids in terms of height and width. The answers given by individuals before a certain stage of development are the result of the "height" agent winning over the "width" agent. After a certain stage, a middle-manager agent exists which neutralizes the competing answers, allowing the individual to say they are the same. In its broadest terms, the problem left is how these agents ever come to compete with each other.

This thesis outlines a theory of the genesis of the conflict and the resulting middle-manager agent. In this theory, weaker agents are strengthened in other contexts--contexts where they are not dominated. Therefore, I propose that a central part of the transition between different ways of knowing is the result of actions in one context, the movement of seemingly unrelated activities (without necessarily having an explicitly formulated goal in doing so), and then the return to the original situations. Individuals who assert that the taller liquid is more do not formulate the situation in terms of a problem to be solved; they are typically quite content with their assessment of which container has more.

Thus, goal-directed activities are less important than are "non-instrumental" activities (such as play) to the emergence of middle-managers. This is new way of thinking about the central importance of behavior which until now has been considered to have little cognitive utility. (Copies available exclusively from MIT **Libraries** , Rm. 14-0551, Cambridge, MA 02139-4307. Ph. 617-253-5668; Fax 617-253-1690.)

11/5/7 (Item 1 from file: 2)
DIALOG(R)File 2:INSPEC

5390278 INSPEC Abstract Number: C9611-6130-013
Title: Clustering **for contingency tables: boxes and partitions**
Author(s): Mirkin, B.
Author Affiliation: DIMACS, Rutgers Univ., Piscataway, NJ, USA
Journal: Statistics and Computing vol.6, no.3 p.217-29
Publisher: Chapman & Hall,
Publication Date: Sept. 1996 Country of Publication: UK
CODEN: STACE3 ISSN: 0960-3174
SICI: 0960-3174(199609)6:3L.217:CCTB;1-Z
Material Identity Number: P702-96003
Language: English Document Type: Journal Paper (JP)
Treatment: Practical (P); Theoretical (T)
Abstract: The correspondence analysis (CA) method appears to be an effective tool for analysis of interrelations between rows and columns in **two** -way contingency data. A discrete **version** of the method, **box clustering** , is developed in the paper using an approximation version of the CA model extended to the case when CA factor values are required to be Boolean. Several properties of the proposed SEFIT-BOX algorithm are proved to facilitate interpretation of its output. It is also shown that two known partitioning algorithms (applied within row or column sets only) could be considered as locally optimal algorithms for fitting the model, and extensions of these algorithms to a simultaneous row and column partitioning problem are proposed. (26 Refs)
Subfile: C
Descriptors: data analysis; probability; statistical analysis
Identifiers: contingency tables; box **clustering** ; correspondence analysis method; two-way contingency data analysis; rows; columns; data interrelations; CA model approximation; Boolean values; SEFIT-BOX algorithm ; partitioning algorithms; locally optimal algorithms; model fitting; sequential fitting; probability
Class Codes: C6130 (Data handling techniques); C1140 (Probability and statistics)

11/5/8 (Item 2 from file: 2)
DIALOG(R)File 2:INSPEC

04213696 INSPEC Abstract Number: B9209-6210-049
Title: Design of leaky bucket access control schemes in ATM networks
Author(s): Chao, H.J.
Author Affiliation: Bellcore, Red Bank, NJ, USA
Conference Title: ICC 91. International Conference on Communications Conference Record (Cat. No.91CH2984-3) p.180-7 vol.1
Publisher: IEEE, New York, NY, USA
Publication Date: 1991 Country of Publication: USA 3 vol. xxix + 1755 pp.
ISBN: 0 7803 0006 8
U.S. Copyright Clearance Center Code: CH2984-3/91/0000-0180$01.00
Conference Sponsor: IEEE
Conference Date: 23-26 June 1991 Conference Location: Denver, CO, USA
Language: English Document Type: Conference Paper (PA)
Treatment: Practical (P)
Abstract: Depending on what enforcement action is taken and whether or not there is a 'shaping' **buffer** , four **versions** of the leaky **bucket**

scheme are considered, and their cell loss performance is compared in conjunction with a statistical multiplexer. Based on the best-performing version, three architectures are proposed. Among them, a novel algorithm and its implementation method are proposed to accommodate a large number of virtual channel connections on each incoming STS-3c channel. A VLSI chip (called a sequencer), containing about 200 K CMOS transistors, has been designed to implement the architecture. (25 Refs)
Subfile: B
Descriptors: CMOS integrated circuits; telecommunication networks; time division multiplexing; VLSI
Identifiers: shaping **buffer** ; B-ISDN; leaky bucket access control; ATM networks; cell loss performance; statistical multiplexer; virtual channel connections; VLSI chip; sequencer; CMOS transistors
Class Codes: B6210 (Telecommunication applications); B6150C (Switching theory); B2570D (CMOS integrated circuits)

11/5/9 (Item 3 from file: 2)
DIALOG(R)File 2:INSPEC

04116034 INSPEC Abstract Number: C9205-6110J-005
Title: The object-based class library **in Borland C++**
Author(s): Horstmann, C.
Journal: C++ Report vol.4, no.1 p.15-18
Publication Date: Jan. 1992 Country of Publication: USA
CODEN: CRPTE7 ISSN: 1040-6042
Language: English Document Type: Journal Paper (JP)
Treatment: Practical (P)
Abstract: Borland C++ **Version** 3.0 includes **two** class **libraries** , principally implementing **containers** . The first one, the so-called object-based **library** , is discussed. It has been supplied with Borland C++ Version 2.0 and slightly enhanced in the newest release. The second **library** is new to Borland C++ 3.0 and is template based. Both **libraries** are included in the professional version of the compiler. All objects in the object-based **library** are derived from an abstract base class Object. The containers are heterogeneous and can hold any collection of instances of classes derived from Object. This setup is commonly known as the Smalltalk approach and is used in many C++ class **libraries** . To test how easy it is to use the **library** , the author modified a project from a C++ training course: a simulation of customers entering a bank and being serviced by an array of tellers. This simulation program has a queue holding customers, an array holding tellers, and a priority queue holding events sorted by a time stamp. Thus, there are naturally three containers in this program. (3 Refs)
Subfile: C
Descriptors: C language; C listings; object-oriented programming
Identifiers: object-based class **library** ; Borland C++ Version 3.0; containers; compiler; abstract base class; Smalltalk approach; simulation; queue; array; priority queue; events; sorted; time stamp
Class Codes: C6110J (Object-oriented programming)

11/5/10 (Item 1 from file: 94)
DIALOG(R)File 94:JICST-EPlus

04561445 JICST ACCESSION NUMBER: 00A0464808 FILE SEGMENT: JICST-E
Speedup of the noise prediction computer system.
ICHIHARA KIYOSHI (1)
(1) Mitsubishi Res. Inst., Inc.
Mitsubishi Sogo Kenkyujo Shoho(Journal of Mitsubishi Research Institute), 2000, NO.36, PAGE.52-73, FIG.9, TBL.7, REF.17
JOURNAL NUMBER: S0866AAY ISSN NO: 0287-2129
UNIVERSAL DECIMAL CLASSIFICATION: 628.517 681.3.02.002
LANGUAGE: Japanese COUNTRY OF PUBLICATION: Japan
DOCUMENT TYPE: Journal
ARTICLE TYPE: Original paper

MEDIA TYPE: Printed Publication
ABSTRACT: The noise prediction computer system("this system") which can support the anti-noise measure planning of thermal power stations has been utilized for 20 years on large computers in Tohoku Electric Power Company, Inc. The recent personal computers'("PC") speedup and operational facility made it possible to immigrate this system from large computers to PC's and also to raise the upper limit of the calculation model scale. As large calculation models might require highspeed processing, the author speeded up this system about three times on one CPU by the following improvements. The first is that the intermediately calculated values are stored in arrays on memories instead of files on hard disks as the **previous version** of this system, and the **second** is that the bounding **box** simple judging method is used to know whether a building might block off the sound line from a source point to an observation one. Further more, for the case of large calculation model, the third method of parallel computing on PC **cluster** (i.e. networked PC's) as max as four CPU's can get a high parallelization efficiency, that is, the execution time is nearly inversely proportional to the CPU numbers. (author abst.)
DESCRIPTORS: industrial noise; noise level; noise control; forecast; speedup; FORTRAN; parallel arithmetic; power plant; computer application system; downsizing
BROADER DESCRIPTORS: noise(pollution); sound; annoyance; environmental pollution control; countermeasure; suppression and control; modification; improvement; high level language; programming language; formal language; language; arithmetic system; method; electric power facility; system; computer system(hardware)
CLASSIFICATION CODE(S): SB04030E; JD02020C

11/5/14 (Item 2 from file: 6)
DIALOG(R)File 6:NTIS

1736045 NTIS Accession Number: AD-M000 175/0
SF 298 Report Documentation Page (for Microcomputers)
(Data file)
Naval Research Lab., Washington, DC.
Corp. Source Codes: 000927000
Report No.: DOD/DF/DK-93/058
Oct 92 diskette
Languages: English
Journal Announcement: GRAI9317
System: IBM-PC compatible; MS DOS operating system.
The datafile is on one 5 1/4 inch diskette, 360K double density. File format: Word Perfect 5.1 or greater. Order this product from NTIS by: phone at 1-800-553-NTIS (U.S. customers); (703)605-6000 (other countries); fax at (703)321-8547; and email at orders@ntis.fedworld.gov. NTIS is located at 5285 Port Royal Road, Springfield, VA, 22161, USA.
NTIS Prices: CP D01
Country of Publication: United States
The Naval Research Laboratory produced the SF 298 in WordPerfect 5.1 to aid in publishing technical reports. The form is set up as a table. There are **two versions** : the first has text markers in the data **boxes** to aid in locating where to put data; the second has no markers.
Descriptors: *Forms(Papers); *Standardization; *Datafile; Printers(Data processing); Military publications
Identifiers: Standard form 298; SF 298; NTISDODN
Section Headings: 88B (Library and Information Sciences--Information Systems); 74E (Military Sciences--Logistics, Military Facilities, and Supplies)

11/5/15 (Item 1 from file: 144)
DIALOG(R)File 144:Pascal

16186845 PASCAL No.: 03-0345159

Performance and scalability of EJB applications

Proceedings of the 2002 ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'02)

CECCHET Emmanuel; MARGUERITE Julie; ZWAENEPOEL Willy
Rice University/INRIA, 655, avenue de l'Europe, 38330 Montbonnot, France; Rice University, 6100 Main Street, 132, Houston, TX, 77005, United States

Association for Computing Machinery (ACM). Special Interest Group on Programming Languages (SIGPLAN), New York, New York, United States; Association for Computing Machinery (ACM). SIGSOFT, New York, New York, United States
OOPSLA'02 ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications, 17 (Seattle, Washington USA) 2002-11-04
Journal: ACM SIGPLAN notices, 2002, 37 (11) 246-261
ISSN: 1523-2867 Availability: INIST-16907; 354000104236640170
No. of Refs.: 27 ref.
Document Type: P (Serial); C (Conference Proceedings) ; A (Analytic)
Country of Publication: United States
Language: English

We investigate the combined effect of application implementation method, container design, and efficiency of communication layers on the performance scalability of J2EE application servers by detailed measurement and profiling of an auction site server. We have implemented five versions of the auction site. The first version uses stateless session beans, making only minimal use of the services provided by the Enterprise JavaBeans (EJB) **container** . **Two versions** use entity beans, one with **container** -managed persistence and the other with bean-managed persistence. The fourth version applies the session facade pattern, using session beans as a facade to access entity beans. The last version uses EJB 2.0 local interfaces with the session facade pattern. We evaluate these different implementations on two popular open-source EJB containers with orthogonal designs. JBoss uses dynamic proxies to generate the container classes at run time, making an extensive use of reflection. JOnAS precompiles classes during deployment, minimizing the use of reflection at run time. We also evaluate the communication optimizations provided by each of these EJB containers. The most important factor in determining performance is the application implementation method. EJB applications with session beans perform as well as a Java servlets-only implementation and an order-of-magnitude better than most of the implementations based on entity beans. The fine-granularity access exposed by the entity beans limits scalability. Use of session facade beans improves performance for entity beans, but only if local communication is very efficient or EJB 2.0 local interfaces are used. Otherwise, session facade beans degrade performance. For the implementation using session beans, communication cost forms the major component of the execution time on the EJB server. The design of the container has little effect on performance. With entity beans, the design of the container becomes important. In particular, the cost of reflection affects performance. For implementations using session facade beans, local communication cost is critically important. EJB 2.0 local interfaces improve the performance by avoiding the communication layers for local communications.

English Descriptors: Occupation time; Scalability; System performance; Reliability; Measurement method; **Cache** memory; JAVA language; Communication; Optimization; Web site; Computer server

French Descriptors: Temps occupation; Extensibilite; Performance systeme; Fiabilite; Methode mesure; Antememoire; Langage JAVA; Communication; Optimisation; Site Web; Serveur informatique; Enterprise JavaBeans (EJB) container design

Classification Codes: 001D02B10; 001D02B03

11/5/16 (Item 2 from file: 144)
DIALOG(R)File 144:Pascal

15643176 PASCAL No.: 02-0348425

The hidden number problem in extension fields and its applications

LATIN 2002 : theoretical informatics : Cancun, 3-6 April 2002

GONZALEZ VASCO Maria Isabel; NAESLUND Mats; SHPARLINSKI Igor E

RAJSBAUM Sergio, ed

Dept. of Mathematics, University of Oviedo, Oviedo 33007, Spain; Ericsson Research, 16480 Stockholm, Sweden; Dept. of Computing, Macquarie University , Sydney, NSW 2109, Australia

Latin American symposium, 5 (Cancun MEX) 2002-04-03

Journal: Lecture notes in computer science, 2002, 2286 105-117

ISBN: 3-540-43400-3 ISSN: 0302-9743 Availability: INIST-16343; 354000097090400080

No. of Refs.: 38 ref.

Document Type: P (Serial); C (Conference Proceedings) ; A (Analytic)

Country of Publication: Germany

Language: English

We present polynomial time algorithms for certain generalizations of the hidden number problem which has played an important role in gaining understanding of the security of commonly suggested one way functions. Namely, we consider an analogue of this problem for a certain class of polynomials over an extension of a finite field; recovering a hidden polynomial given the values of its trace at randomly selected points. Also, we give an algorithm for a variant of the problem in free finite dimensional modules. This result can be helpful for studying security of analogues of the RSA and Diffie-Hellman cryptosystems over such modules. The hidden number problem is also related to the so called black- **box** field model of computation. We show that simplified **versions** of the above recovery problems can be used to derive positive results on the computational power of this model.

```
File 347:JAPIO Nov 1976-2004/Aug(Updated 041203)
         (c) 2004 JPO & JAPIO
File 350:Derwent WPIX 1963-2005/UD,UM &UP=200501
         (c) 2005  Thomson Derwent

Set     Items   Description
S1     268831   CACH??? OR BUFFER???
S2     907531   BIN OR BINS OR BOX OR BOXES OR CONTAINER? ? OR BUCKET? ?
S3       1318   (PAST OR OLD??? OR PRIOR? OR PREVIOUS? OR PRECED??? OR EAR-
             LY OR EARLIE?? OR FORMER OR AGED OR AGING OR HISTOR???)(7N)(V-
             ERSION? ? OR EDITION? ?)
S4       5678   (DIFFERENT OR VARIOUS OR SEPARATE OR ALTERNAT??? OR SECOND
             OR 2ND OR TWO OR MULTIPLE OR MULTIPLICITY OR SEVERAL OR MANY -
             OR PLURAL? OR ASSORT???? OR ADDITIONAL)(7N)(VERSION? ? OR EDI-
             TION? ?)
S5         62   S2(10N)S3:S4
S6          1   S1 AND S5
S7        231   S2(10N)(VERSION? ? OR EDITION? ?)
S8          9   S1 AND S7
S9          1   S5 AND (DATABASE? ? OR DATA()BASE? ? OR DBMS OR RDBMS OR R-
             EPOSITOR??? OR LIBRAR??? OR ARCHIVE? ? OR (INFORMATION OR DAT-
             A)()MANAG??? OR DATE()(MART? ? OR WAREHOUSE? ?) OR DATAMART? ?
              OR CLUSTER???)
S10         9   S6 OR S8:S9
S11         5   S10 AND IC=G06F
S12         4   S10 NOT S11
```

11/5/1 (Item 1 from file: 347)
DIALOG(R)File 347:JAPIO

03946142 **Image available**
OPERATING MANUAL PREPARING DEVICE

PUB. NO.: 04-311242 [JP 4311242 A]
PUBLISHED: November 04, 1992 (19921104)
INVENTOR(s): MOTOHASHI MASAAKI
APPLICANT(s): RICOH CO LTD [000674] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 03-076613 [JP 9176613]
FILED: April 10, 1991 (19910410)
INTL CLASS: [5] **G06F-013/00** ; **G06F-003/02** ; **G06F-009/06**
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units); 44.7 (COMMUNICATION -- Facsimile); 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units); 45.3 (INFORMATION PROCESSING -- Input Output Units)
JAPIO KEYWORD:R107 (INFORMATION PROCESSING -- OCR & OMR Optical Readers)
JOURNAL: Section: P, Section No. 1504, Vol. 17, No. 132, Pg. 131, March 18, 1993 (19930318)

ABSTRACT

PURPOSE: To simplify the accessing procedure for receiving mail by preparing an operating manual for receiving mail service and transmitting the manual at the time of transmitting a mail acknowledgment to the transmitting address of the mail.

CONSTITUTION: When a communication control section 13 receives a confidential mail service request or post-office **box** mail service request requesting the output of an acknowledgment, an **edition** control section 23 edits and prepares an operating manual explaining the operating method for fetching the mail accumulated in facsimile mail equipment 1 from communication terminal by expanding the manual in a manual preparing **buffer** 14 and transmits the prepared manual to the communication equipment at the transmitting address of the mail as the operating manual for receiving the mail when the section 23 transmits the acknowledgment of the mail.

11/5/2 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

015661228 **Image available**
WPI Acc No: 2003-723415/200369
XRPX Acc No: N03-578419
Electronic mail server operating method for wireless message client devices involves sending message to message client device associated with mail box **to allow client device to synchronize** cached version
Patent Assignee: OPENWAVE SYSTEMS INC (OPEN-N); BROWN B L (BROW-I); ELLSWORTH B J S (ELLS-I)
Inventor: BROWN B L; ELLSWORTH B J S
Number of Countries: 031 Number of Patents: 002
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 1343103	A2	20030910	EP 2002259041	A	20021231	200369	B
US 20030177171	A1	20030918	US 200254771	A	20020122	200369	

Priority Applications (No Type Date): US 200254771 A 20020122
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 1343103	A2	E	14	G06F-017/60	

Designated States (Regional): AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI SK TR

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20030177171	A1			G06F-015/16	

Abstract (Basic): EP 1343103 A2

NOVELTY - The mail server system has mail boxes associated with message client devices (12). Changes in organizational structure of mail box is carried out based on an input change request. A message is sent to a message client device associated with the mail **box** , to allow message client device to synchronize a **cached** **version** of mail **box** stored locally.

DETAILED DESCRIPTION - INDEPENDENT CLAIMs are also included for the following:

(1) A method of operating a message client device;

(2) An electronic mail server system; and

(3) A message client device.

USE - For mobile or wireless message client devices such as mobile telephones, Personal Digital Assistant (PDA), pocket PCs, Notebook computers etc.

ADVANTAGE - Allows users to view new mail instantaneously without having to wait for message client device to download new mail.

DESCRIPTION OF DRAWING(S) - The figure illustrates operations that occur when an external MTA server sends a message to an e-mail server system.

Message client devices. (12)

pp; 14 DwgNo 4/7

Title Terms: ELECTRONIC; MAIL; SERVE; OPERATE; METHOD; WIRELESS; MESSAGE; CLIENT; DEVICE; SEND; MESSAGE; MESSAGE; CLIENT; DEVICE; ASSOCIATE; MAIL; BOX; ALLOW; CLIENT; DEVICE; SYNCHRONISATION; VERSION

Derwent Class: T01; W01

International Patent Class (Main): **G06F-015/16** ; **G06F-017/60**

File Segment: EPI

11/5/3 (Item 2 from file: 350)

DIALOG(R)File 350:Derwent WPIX

014825015 **Image available**

WPI Acc No: 2002-645721/200270

XRPX Acc No: N02-510524

Data objects downloading device e.g. set top box **overwrites stored module memory with current** version **when stored module is not current version upon receiving object request**

Patent Assignee: SONY SERVICE CENT EURO NV (SONY); SKARINGER L (SKAR-I); THIERY O (THIE-I)

Inventor: SKARINGER L; THIERY O

Number of Countries: 030 Number of Patents: 005

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
EP 1227667	A1	20020731	EP 2001300430	A	20010118	200270	B
US 20020107995	A1	20020808	US 200252860	A	20020116	200270	
CN 1366244	A	20020828	CN 2002102374	A	20020118	200282	
JP 2002314895	A	20021025	JP 200210801	A	20020118	200303	
KR 2002062184	A	20020725	KR 20022661	A	20020117	200308	

Priority Applications (No Type Date): EP 2001300430 A 20010118

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 1227667	A1	E	13	H04N-005/00	

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20020107995	A1			G06F-009/44	
CN 1366244	A			G06F-012/06	
JP 2002314895	A		9	H04N-005/44	
KR 2002062184	A			H04N-007/12	

Abstract (Basic): EP 1227667 A1

NOVELTY - An object layer interface (12) downloads the current version of a stored module in a memory (14) and overwrites the module when the stored module is not the current version upon receiving a request for object from an application (10). A controller (18) copies

into an object **cache** (20) only the objects of the stored module still in use by the application before the interface overwrites.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

(1) Method of providing downloaded data objects to an application; and

(2) Computer readable storage medium storing downloaded data objects providing program.

USE - E.g. set top box that conforms to MHP specifications for downloading data objects such as DSM-CC objects.

ADVANTAGE - By updating the module with the current version, future access times are reduced. Since only the latest available version is kept in the memory at any given time, the memory efficiency is improved.

DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of the data objects downloading device.

Application (10)
Object layer interface (12)
Memory (14)
Controller (18)
Object **cache** (20)
pp; 13 DwgNo 2/4

Title Terms: DATA; OBJECT; DEVICE; SET; TOP; BOX; STORAGE; MODULE; MEMORY; CURRENT; VERSION; STORAGE; MODULE; CURRENT; VERSION; RECEIVE; OBJECT; REQUEST
Derwent Class: T01; W03; W04
International Patent Class (Main): **G06F-009/44** ; **G06F-012/06** ; H04N-005/00; H04N-005/44; H04N-007/12
International Patent Class (Additional): H04H-001/00
File Segment: EPI

11/5/4 (Item 3 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014788501 **Image available**
WPI Acc No: 2002-609207/200265
Related WPI Acc No: 1999-494629; 2002-339699; 2002-636953; 2002-666065
XRPX Acc No: N02-482415

Managing system caches **by sending write notification message from master to node indicating persistent storage of most recent modified data item version**

Patent Assignee: ORACLE INT CORP (ORAC-N); ORACLE CORP (ORAC-N); BAMFORD R J (BAMF-I); BRIDGE W H (BRID-I); BROWER D (BROW-I); CHAN W W S (CHAN-I); CHANDRASEKARAN S (CHAN-I); MACNAUGHTON N (MACN-I); SRIHARI V (SRIH-I)
Inventor: BAMFORD R; BRIDGE W; BROWER D; CHAN W; CHANDRASEKARAN S; MACNAUGHTON N; SRIHARI V; BAMFORD R J; BRIDGE W H; CHAN W W S
Number of Countries: 101 Number of Patents: 006
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
WO 200271229	A2	20020912	WO 2002US7475	A	20020306	200265	B
US 20020095403	A1	20020718	US 98199120	A	19981124	200271	
			US 2001274270	P	20010307		
			US 200291618	A	20020304		
EP 1366420	A2	20031203	EP 2002748408	A	20020306	200380	
			WO 2002US7475	A	20020306		
AU 2002248570	A1	20020924	AU 2002248570	A	20020307	200433	
AU 2002335503	A1	20020919	AU 2002335503	A	20020306	200433	
CN 1496510	A	20040512	CN 2002806162	A	20020306	200452	
			WO 2002US7475	A	20020306		

Priority Applications (No Type Date): US 200291618 A 20020304; US 2001274270 P 20010307; US 98199120 A 19981124; US 200292247 A 20020304
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 200271229	A2	E	64	G06F-012/00	

Designated States (National): AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA

CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN
IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ
OM PH PL PT RO RU SD SE SG SI SK SL TJ TM TN TR TT TZ UA UG UZ VN YU ZA
ZM ZW
Designated States (Regional): AT BE CH CY DE DK EA ES FI FR GB GH GM GR
IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZM ZW

US 20020095403	A1	34	G06F-007/00	CIP of application US 98199120 Provisional application US 2001274270 CIP of patent US 6353836
EP 1366420	A2 E		G06F-012/08	Based on patent WO 200271229

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT
LI LT LU LV MC MK NL PT RO SE SI TR

AU 2002248570	A1	G06F-012/00	Based on patent WO 200273416
AU 2002335503	A1	G06F-012/00	Based on patent WO 200271229
CN 1496510	A	G06F-012/08	

Abstract (Basic): WO 200271229 A2

NOVELTY - Method for managing **caches** in a multiple **cache** system consists in modifying the data item in a first node of the multiple **caches** to create a modified data item, sending the modified item from the first node to the second node without durably storing the modified item, and then the first node sending a request to a master of the data item for writing it to persistent storage. In response to the request the master alternates with the multiple **caches** to write the item to persistent storage. The method includes maintaining entries for **past** image **versions** of data items within an ordered series of **bins** , each **bin** corresponding to a time range.

DETAILED DESCRIPTION - The request is sent to the master by sending it to a global lock manager which is one manager within a distributed lock management system. Write notification messages are sent from the master to interested nodes indicating that the data item has been written to persistent storage. Co-ordination is by selecting a particular node of the multiple **caches** that has a particular version of the data item which is at least as recent as the modified data item in the first node so that this particular version of the data item is written from that node to persistent storage. There are INDEPENDENT CLAIMS for:

(1) A method of managing a data item
(2) A method of managing versions of a data item
(3) A method for managing past images of a data item
(4) A computer program for **cache** management
(5) A computer program for data item management

USE - Method is for managing **caches** in a system with multiple **caches** that may contain different copies of a data item i.e. to coordinate writing of dirty data items in **databases** .

ADVANTAGE - Method enables dirty versions of the same data item to reside in more than one volatile memory.

DESCRIPTION OF DRAWING(S) - The figure shows how write-do-disk operations are coordinated in a role-based write approach when the request is not from the exclusive lock holder.

pp; 64 DwgNo 4d/9

Title Terms: MANAGE; SYSTEM; SEND; WRITING; NOTIFICATION; MESSAGE; MASTER; NODE; INDICATE; PERSISTENT; STORAGE; RECENT; MODIFIED; DATA; ITEM; VERSION
Derwent Class: T01
International Patent Class (Main): **G06F-007/00** ; **G06F-012/00** ; **G06F-012/08**
International Patent Class (Additional): **G06F-011/14** ; **G06F-017/30**
File Segment: EPI

11/5/5 (Item 4 from file: 350)
DIALOG(R)File 350:Derwent WPIX

013661732 **Image available**
WPI Acc No: 2001-145944/200115
XRPX Acc No: N01-106726

Information amount reduction from server in communication system, by sending new version of named container file to client extension, if catalog whose identifier matches with that from client is not in catalog list

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: BITTINGER R; BRUBAKER N C; HOUSEL B C; WANG S
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6148340	A	20001114	US 9870513	A	19980430	200115	B

Priority Applications (No Type Date): US 9870513 A 19980430
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6148340	A		11	G06F-003/00	

Abstract (Basic): US 6148340 A

NOVELTY - A catalog created from new **version** of named **container** file received from a web server, is added in a catalog list. A new **version** of named **container** file is sent to the client extension, when catalog whose identifier matches with that received from client is not found in the catalog list.

DETAILED DESCRIPTION - The client and server utilize containers for transmission of information. Each container has one or more objects having associated lengths, cyclic redundant checks, header and content portion. An uniform resource locator consisting of name of a container file, requested by a browser at the client is passed to client extension. The requested name of the container file as one that had been previously **cached** is recognized by the client extension. Request including unique identifier of the previously **cached** container having the requested name is passed to a server extension, which is then sent to server. INDEPENDENT CLAIMS are also included for the following:

(a) information amount reducing method from server;

(b) information amount reducing program from server

USE - For reducing amount of information sent from server to client in computer network communication system.

ADVANTAGE - Reduces the time required to maintain currency of container objects in distributed systems. Avoids UNZIP processing to determine if objects require updating at the client. Reduces overall network traffic and increases the reliability of information transmitted.

DESCRIPTION OF DRAWING(S) - The figure shows the graphical representation of client-server connection.

```
File 348:EUROPEAN PATENTS 1978-2004/Dec W04
         (c) 2005 European Patent Office
File 349:PCT FULLTEXT 1979-2002/UB=20041230,UT=20041223
         (c) 2004 WIPO/Univentio

Set     Items   Description
S1     282169   CACH??? OR BUFFER???
S2     439253   BIN OR BINS OR BOX OR BOXES OR CONTAINER? ? OR BUCKET? ?
S3      10067   (PAST OR OLD??? OR PRIOR? OR PREVIOUS? OR PRECED??? OR EAR-
             LY OR EARLIE?? OR FORMER OR AGED OR AGING OR HISTOR???)(7N)(V-
             ERSION? ? OR EDITION? ?)
S4      55006   (DIFFERENT OR VARIOUS OR SEPARATE OR ALTERNAT??? OR SECOND
             OR 2ND OR TWO OR MULTIPLE OR MULTIPLICITY OR SEVERAL OR MANY -
             OR PLURAL? OR ASSORT???? OR ADDITIONAL)(7N)(VERSION? ? OR EDI-
             TION? ?)
S5        516   S2(10N)S3:S4
S6          9   S1(50N)S5
S7       1921   S2(10N)(VERSION? ? OR EDITION? ?)
S8         28   S1(50N)S7
S9         32   S5(50N)(DATABASE? ? OR DATA()BASE? ? OR DBMS OR RDBMS OR R-
             EPOSITOR??? OR LIBRAR??? OR ARCHIVE? ? OR (INFORMATION OR DAT-
             A)()MANAG??? OR DATE()(MART? ? OR WAREHOUSE? ?) OR DATAMART? ?
              OR CLUSTER???)
S10         0   S5(50N)(DATA()(MART? ? OR WAREHOUSE? ?))
S11        63   S6 OR S8 OR S9
S12        55   S11 AND AC=US/PR
S13        28   S12 AND AY=(1970:1998)/PR
S14        19   S11 AND PY=1970:1998
S15        31   S13:S14
S16        14   S15 AND IC=G06F
```

16/3,K/2 (Item 2 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

01052722
SOFTWARE UPDATE MANAGER
VERWALTUNGSVORRICHTUNG ZUR SOFTWARE-AKTUALISIERUNG
GESTIONNAIRE DE MISE A JOUR DE LOGICIELS
PATENT ASSIGNEE:
Koninklijke Philips Electronics N.V., (200769), Groenewoudseweg 1, 5621 BA Eindhoven, (NL), (Proprietor designated states: all)
INVENTOR:
REHA, Mark, Keith, Prof. Holstlaan 6, NL-5656 AA Eindhoven, (NL)
MORRIS, Charles, Prof. Holstlaan 6, NL-5656 AA Eindhoven, (NL)
LEGAL REPRESENTATIVE:
Faessen, Louis Marie Hubertus (19891), INTERNATIONAAL OCTROOIBUREAU B.V., Prof. Holstlaan 6, 5656 AA Eindhoven, (NL)
PATENT (CC, No, Kind, Date): EP 951679 A1 991027 (Basic)
EP 951679 B1 030416
WO 99024945 990520
APPLICATION (CC, No, Date): EP 98939808 980908; WO 98IB1392 980908
PRIORITY (CC, No, Date): US 968020 971112
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS: **G06F-013/00**
NOTE:
No A-document published by EPO
LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	200316	487
CLAIMS B	(German)	200316	475
CLAIMS B	(French)	200316	567
SPEC B	(English)	200316	5234
Total word count - document A			0
Total word count - document B			6763
Total word count - documents A + B			6763

INTERNATIONAL PATENT CLASS: **G06F-013/00**

...SPECIFICATION may go back to the previous version of the display driver, or even go back to any **earlier version** . Downloading **previous versions** is accomplished by selecting the desired **version** from software component update list **box** 60. The **archived** version is then retrieved from software component storage 22 and downloaded to the user. What's New...

16/3,K/3 (Item 3 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00657924
Modem with firmware upgrade feature and system and method therefor
Modem mit Firmware-Aktualisierungsmoglichkeit, entsprechendes System und Verfahren
Modem avec la possibilite de mise-a-niveau du microprogramme, systeme et procede correspondants
PATENT ASSIGNEE:
Multi-Tech Systems Inc, (1808070), 2205 Woodale Drive, Mounds View, Minnesota 55112, (US), (Proprietor designated states: all)
INVENTOR:
Johnson, Greg, 1505 Ferndale Avenue, Fridley, Minnesota 55432, (US)
Weinzierl, David A., 7661 Greenfield Avenue N.E., Apt 306, Mounds View, Minnesota 55112, (US)
Johnson, Richard D., 1695 East County Road D, No 301, Maplewood, Minnesota 55109, (US)
LEGAL REPRESENTATIVE:
McDonough, Jonathan et al (94601), Urquhart-Dykes & Lord LLP Tower North

Central Merrion Way, Leeds LS2 8PA, (GB)
PATENT (CC, No, Kind, Date): EP 632629 B1 040922 (Basic)
APPLICATION (CC, No, Date): EP 94304741 940629;
PRIORITY (CC, No, Date): US 87164 930702
DESIGNATED STATES: AT; BE; CH; DE; DK; ES; FR; GB; GR; IE; IT; LI; LU; MC; NL; PT; SE
INTERNATIONAL PATENT CLASS: H04L-029/06; **G06F-009/445** ; H04M-011/06
ABSTRACT WORD COUNT: 113
NOTE:
Figure number on first page: 2

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF2	1122
CLAIMS B	(English)	200439	989
CLAIMS B	(German)	200439	894
CLAIMS B	(French)	200439	1221
SPEC A	(English)	EPABF2	6991
SPEC B	(English)	200439	7273
Total word count - document A			8114
Total word count - document B			10377
Total word count - documents A + B			18491

...INTERNATIONAL PATENT CLASS: **G06F-009/445**
...SPECIFICATION on the version number received. The host PC then sets the max packet size according to the **version** number received.
Next, as shown in control **box** 880 in the top right of Figure 8C, the host PC initializes pointers to the top of the RAM **buffer** which was allocated in control box 804 shown on Figure 8A, and in which the processed and...

...SPECIFICATION on the version number received. The host PC then sets the max packet size according to the **version** number received.
Next, as shown in control **box** 880 in the top right of Figure 8C, the host PC initializes pointers to the top of the RAM **buffer** which was allocated in control box 804 shown on Figure 8A, and in which the processed and...

16/3,K/4 (Item 4 from file: 348)
DIALOG(R)File 348:EUROPEAN PATENTS

00578607
Method of reproducing text on a raster output device
Verfahren zur Wiedergabe von Text auf einem Rasterausgangsgerat
Methode de reproduction de texte sur un dispositif de sortie a balayage de trames
PATENT ASSIGNEE:
Oce-Technologies B.V., (241033), St. Urbanusweg 43, 5914 CC Venlo, (NL), (applicant designated states: DE;FR;GB;IT;NL)
INVENTOR:
Koopman, Stephan, Bernardus, Jozef, Kesselseweg 30, 5953 CL Reuver, (NL)
Smeijers, Fredericus, Ronaldus, Johannes, Amsterdamseweg 21, 6814 GA Arnhem, (NL)
LEGAL REPRESENTATIVE:
Hanneman, Henri W., Dr. et al (49472), Oce-Technologies B.V. Patents & Information St. Urbanusweg 43 P.O. Box 101, 5900 MA Venlo, (NL)
PATENT (CC, No, Kind, Date): EP 579873 A1 940126 (Basic)
EP 579873 B1 990506
APPLICATION (CC, No, Date): EP 92202211 920720;
PRIORITY (CC, No, Date): EP 92202211 920720
DESIGNATED STATES: DE; FR; GB; IT; NL
INTERNATIONAL PATENT CLASS: **G06F-003/153** ; G06K-015/12; G09G-005/28
ABSTRACT WORD COUNT: 41

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9918	586
CLAIMS B	(German)	9918	526
CLAIMS B	(French)	9918	620
SPEC B	(English)	9918	3086
Total word count - document A			0
Total word count - document B			4818
Total word count - documents A + B			4818

INTERNATIONAL PATENT CLASS: **G06F-003/153** ...

...SPECIFICATION the right-hand side of the character, which functions as the character white area when two character **boxes** are assembled to form words. The **second** font **version** differs from the first version in that the rightmost column of pixels (white in the first **version**) of the character **boxes** is filled with grey pixels. The horizontal extent of the character boxes is shown in Fig.2 by reference sign 14.

In operation, text data are delivered serially by source 51 to **buffer** 54, which in turn delivers the data to **buffer** 55 one processing cycle later. This way, if **buffer** 55 contains the code for a particular character of the text, referred to as "current character" hereafter...

16/3,K/7 (Item 2 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00466800 **Image available**
NETWORK OBJECT CACHE ENGINE
MOTEUR D'ANTEMEMOIRE POUR OBJETS RESEAU
Patent Applicant/Assignee:
CACHEFLOW INC,
MALCOLM Michael,
ZARNKE Robert,
Inventor(s):
MALCOLM Michael,
ZARNKE Robert,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9857265 A1 **19981217**
Application: WO 98US11834 19980609 (PCT/WO US9811834)
Priority Application: US 9748986 19970609
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES FI GB GE GH GM
GW HU ID IL IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD MG MK MN MW MX
NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG US UZ VN YU ZW GH
GM KE LS MW SD SZ UG ZW AM AZ BY KG KZ MD RU TJ TM AT BE CH CY DE DK ES
FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN ML MR NE SN TD
TG
Publication Language: English
Fulltext Word Count: 10554

Patent and Priority Information (Country, Number, Date):
Patent: ... **19981217**
Main International Patent Class: **G06F-011/14**
Fulltext Availability:
Detailed Description
Publication Year: **1998**

Detailed Description
... disk read access times. Moreover, there is no special requirement that the network objects 114 in the **cache** 102 must have unique names: when network objects 114 have identical names (such as when they are of the same network object 114), the hash table 350 simply points to the same hash **bucket** 340 for both of them.

When there are both **old** and new **versions** of the same network object 114.
the **cache** eneine I 00 resolves new references by the URL 3 1 0 only to the new version...

16/3,K/8 (Item 3 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00450343
MIGRATING TO A SUCCESSIVE SOFTWARE DISTRIBUTION LEVEL
PROCEDE DE PASSAGE AU NIVEAU SUIVANT D'UNE REPARTITION LOGICIELLE PRESENTANT DES MODIFICATIONS LOCALES
Patent Applicant/Assignee:
SIEBEL SYSTEMS INC,
BRODERSEN Robert S,
COKER John L,
HASTINGS Mark C,
Inventor(s):
BRODERSEN Robert S,
COKER John L,
HASTINGS Mark C,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9840807 A2 **19980917**
Application: WO 98US3575 19980224 (PCT/WO US9803575)
Priority Application: US 9739467 19970227
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES FI GB GE GH GM GW HU ID IL IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG US UZ VN YU ZW GH GM KE LS MW SD SZ UG ZW AM AZ BY KG KZ MD RU TJ TM AT BE CH DE DK ES FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN ML MR NE SN TD TG
Publication Language: English
Fulltext Word Count: 10969

Patent and Priority Information (Country, Number, Date):
Patent: ... **19980917**
Main International Patent Class: **G06F-009/44**
Fulltext Availability:
Detailed Description
Publication Year: **1998**

Detailed Description
... brings up the Application Upgrader Object List screen in the background. The following table lists the Merge **Repositories** dialog box options.

Merge **Repositories** Dialog Box Options
OPTION DESCRIPTION
Merge Button Initiates the merge or compare process.
Cancel Button Cancels the current merge request and closes the Merge **Repositories** dialog box. The Application Upgrades Object List screen remams.

Advanced Button Brings up the Merge Options dialog **box** . The **box** is described below.

Prior Standard **Prior** Siebel release.

Repository
Prior Customized Customized **version** corresponding to the **Prior** **Repository** Siebel **Repository** .

New Standard New **version** v3.x Siebel release.

Repository
New Customized Final result of the merge process.

Repository
Merge Options Dialog Box
The Merge Options dialog box is used to specify options to modify the merge process. The default settings are recommended settings. As with the Merge **Repositories** dialog box, this dialog box may be accessed by pufl-down menus, buttons, or other means.

Merge...

16/3,K/12 (Item 7 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00300850 **Image available**
UPDATE MECHANISM FOR COMPUTER STORAGE CONTAINER MANAGER
MOYEN DE MISE A JOUR POUR MODULE DE GESTION D'ELEMENTS DE STOCKAGE D'ORDINATEURS
Patent Applicant/Assignee:
APPLE COMPUTER INC,
Inventor(s):
HARRIS Jared M,
RUBEN Ira L,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9519001 A1 **19950713**
Application: WO 95US196 19950104 (PCT/WO US9500196)
Priority Application: US 94177853 19940105
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AM AT AU BB BG BR BY CA CH CN CZ DE DK EE ES FI GB GE HU JP KE KG KP KR
KZ LK LR LT LU LV MD MG MN MW MX NL NO NZ PL PT RO RU SD SE SI SK TJ TT
UA UZ VN KE MW SD SZ AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE BF
BJ CF CG CI CM GA GN ML MR NE SN TD TG
Publication Language: English
Fulltext Word Count: 119635

Patent and Priority Information (Country, Number, Date):
Patent: ... **19950713**
Main International Patent Class: **G06F-009/44**
Fulltext Availability:
Claims
Publication Year: **1995**

Claim
... final bytes in any existing
file,

2 Label flags
unsigned short flags;
These are currently unused.

3 **Buffer** size
unsigned short blockSize;
This defines the size of the TOC blocks in this **container** , in multiples of 1024 bytes.
4 **Container** format major **version** number
unsigned short majorVersion;
The major format **version** number changes only on incompatible format changes.

5 **Container** format minor **version** number
unsigned short minorVersion;
The minor **version** number changes on upward

c atible format changes, When the major version
OMP
number changes, the minor...

16/3,K/13 (Item 8 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00212824
BINARY DATA COMMUNICATION SYSTEM
SYSTEME DE COMMUNICATION DE DONNEES BINAIRES
Patent Applicant/Assignee:
UNISYS CORPORATION,
Inventor(s):
TEAGUE Tommy K,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9210035 A1 **19920611**
Application: WO 91US8878 19911127 (PCT/WO US9108878)
Priority Application: US 90977 19901130
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AT BE CH DE DK ES FR GB GR IT JP LU NL SE
Publication Language: English
Fulltext Word Count: 5925

Patent and Priority Information (Country, Number, Date):
Patent: ... **19920611**
...International Patent Class: **G06F-05:00**
Fulltext Availability:
Detailed Description
Publication Year: **1992**

Detailed Description
```
... output.fileo;
printf("@nEncoding (%d)
"%s
" as
 18%s
H version,,
fin.id, fout.id);
memset(text buffer ,t85[0],TEXT BUFFER SIZE);
fprintf(fout,
"%s td %s %Id %04X 104X
r
nno
id.string,
version,
ffblk.name,
ffblk.size,
ffblk.wr date,
ffblk.wr time
record=l;
switch( version
case 100.

while(fread( bin . bufferoll , BIN . BUFFER .SIZE.100,,fin)>O)
record++;
for(i=O,j=O;i<BIN BUFFER SIZE 100;i+=3,j+=4)
three2four(bin -13uf f er+-11", tex:E.buf f er+j)
cksum=record&Oxff;
for(i=O;i<BIN BUFFER SIZE 100;i++)
cktum=(cksum+i*bin. buffer [i])&Oxff;
if( fwr3..te(text buffer,TEXT BUFFER SIZE 100,1,fout)
== 0 1 1 fprllntf...
```

16/3,K/14 (Item 9 from file: 349)
DIALOG(R)File 349:PCT FULLTEXT

00118778
TIME-ORDERED DATA BASE
BASE DE DONNEES A ORGANISATION TEMPORELLE
Patent Applicant/Assignee:
  WESTERN ELECTRIC COMPANY INC,
Inventor(s):
  BOYLE Gerald Carroll,
Patent and Priority Information (Country, Number, Date):
  Patent: WO 8402023 A1 **19840524**
  Application: WO 83US1697 19831102 (PCT/WO US8301697)
  Priority Application: US 82731 19821115
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
  AT BE CH DE FR GB JP LU NL SE
Publication Language: English
Fulltext Word Count: 7152

Patent and Priority Information (Country, Number, Date):
  Patent: ... **19840524**
Main International Patent Class: **G06F-015/40**
Fulltext Availability:
  Detailed Description
Publication Year: **1984**

Detailed Description
... box 102 of FIG. 9, the time stamp Ti is used to select the version of the **data** **base** which corresponds to the state of the physical assignments at the time Tie This, of course, is the version of the **data** **base** created by the transaction which took place at time Ti.lf lee*, the **data** **base** **version** created to correspond to the next **earlier** point of time.

In **box** 102, the transaction necessary to carry out the reassignment of facilities is executed. It will be noted that the **data** **base** version operated on (version Ti.1) does not represent the current physical assignments of the facilities, but...

Search Results

Search Results for: **[(cach* or buffer*) <near/50> ((bin or bins or box or boxes or container* or bucket*) <near/10> ((past or old* or prior* or previous* or preced* or early or earlie* or former or aged or aging or histor* or different or various or separate or alternat* or second or 2nd or two or multiple or multiplicity or several or many or plural* or assort* or additional) <near/7> (version* or edition*)))]**
Found **4** of **148,162 searched.**

Search within Results

> Advanced Search

> Search Help/Tips

Sort by: Title Publication Publication Date Score Binder

Results 1 - 4 of 4 short listing

1 A program for social science computer literacy 92%
Paul J. Strand
ACM SIGSOC Bulletin , Proceedings of the joint conference on Easier and more productive use of computer systems. (Part - II): Human interface and the user interface - Volume 1981 May 1981
Volume 13 Issue 2-3
A strategy for organizing the social science computer user community is presented. The strategy recognizes that social scientists have exceptional educational needs and unfavorable budgetary constraints. A series of workshops is proposed to reduce curriculum redundancy and avoid the costly "on demand" mode of consultation that has developed in most computer centers. An example of a workshop is provided.

2 Thread scheduling for cache locality 83%
James Philbin , Jan Edler , Otto J. Anshus , Craig C. Douglas , Kai Li
Proceedings of the seventh international conference on Architectural support for programming languages and operating systems October 1996
Volume 30 , 31 Issue 5 , 9
This paper describes a method to improve the cache locality of sequential programs by scheduling fine-grained threads. The algorithm relies upon hints provided at the time of thread creation to determine a thread execution order likely to reduce cache misses. This technique may be particularly valuable when compiler-directed tiling is not feasible. Experiments with several application programs, on two systems with different cache structures, show that our thread scheduling method can improve pro ...

75%

3 Mixing translucent polygons with volumes
Kevin Kreeger , Arie Kaufman
Proceedings of the conference on Visualizati n '99: celebrating ten years October 1999

We present an algorithm which renders opaque and/or translucent polygons embedded within volumetric data. The processing occurs such that all objects are composited in the correct order, by rendering thin slabs of the translucent polygons between volume slices using slice-order volume rendering. We implemented our algorithm with OpenGL on current general-purpose graphics systems. We discuss our system implementation, speed and image quality, as well as the renderings of several mixe ...

4 Page placement algorithms for large real-indexed caches 72%
R. E. Kessler , Mark D. Hill
ACM Transactions on Computer Systems (TOCS) November 1992
Volume 10 Issue 4

When a computer system supports both paged virtual memory and large real-indexed caches, cache performance depends in part on the main memory page placement. To date, most operating systems place pages by selecting an arbitrary page frame from a pool of page frames that have been made available by the page replacement algorithm. We give a simple model that shows that this naive (arbitrary) page placement leads to up to 30% unnecessary cache conflicts. We develop several page placement algor ...

Results 1 - 4 of 4 **short listing**

```
File 275:Gale Group Computer DB(TM) 1983-2005/Jan 06
         (c) 2005 The Gale Group
File 621:Gale Group New Prod.Annou.(R) 1985-2005/Jan 06
         (c) 2005 The Gale Group
File 636:Gale Group Newsletter DB(TM) 1987-2005/Jan 06
         (c) 2005 The Gale Group
File  16:Gale Group PROMT(R) 1990-2005/Jan 06
         (c) 2005 The Gale Group
File 160:Gale Group PROMT(R) 1972-1989
         (c) 1999 The Gale Group
File 148:Gale Group Trade & Industry DB 1976-2005/Jan 06
         (c)2005 The Gale Group
File 624:McGraw-Hill Publications 1985-2005/Jan 05
         (c) 2005 McGraw-Hill Co. Inc
File  15:ABI/Inform(R) 1971-2005/Jan 05
         (c) 2005 ProQuest Info&Learning
File 647:CMP  Computer Fulltext 1988-2005/Dec W3
         (c) 2005 CMP Media, LLC
File 674:Computer News Fulltext 1989-2004/Dec W2
         (c) 2004 IDG Communications
File 696:DIALOG Telecom. Newsletters 1995-2005/Jan 05
         (c) 2005 The Dialog Corp.
File 369:New Scientist 1994-2005/Dec W4
         (c) 2005 Reed Business Information Ltd.
File 810:Business Wire 1986-1999/Feb 28
         (c) 1999 Business Wire
File 813:PR Newswire 1987-1999/Apr 30
         (c) 1999 PR Newswire Association Inc

Set     Items   Description
S1     328572   CACH??? OR BUFFER???
S2    2113871   BIN OR BINS OR BOX OR BOXES OR CONTAINER? ? OR BUCKET? ?
S3     204619   (PAST OR OLD??? OR PRIOR? OR PREVIOUS? OR PRECED??? OR EAR-
             LY OR EARLIE?? OR FORMER OR AGED OR AGING OR HISTOR???)(7N)(V-
             ERSION? ? OR EDITION? ?)
S4     323217   (DIFFERENT OR VARIOUS OR SEPARATE OR ALTERNAT??? OR SECOND
             OR 2ND OR TWO OR MULTIPLE OR MULTIPLICITY OR SEVERAL OR MANY -
             OR PLURAL? OR ASSORT???? OR ADDITIONAL)(7N)(VERSION? ? OR EDI-
             TION? ?)
S5       3547   S2(10N)S3:S4
S6         29   S1(50N)S5
S7         21   RD (unique items)
S8      23045   S2(10N)(VERSION? ? OR EDITION? ?)
S9        198   S1(50N)S8
S10       115   RD (unique items)
S11        85   S10 NOT PY=1999:2004
S12        75   S11 NOT S7
S13       250   S5(50N)(DATABASE? ? OR DATA()BASE? ? OR DBMS OR RDBMS OR R-
             EPOSITOR??? OR LIBRAR??? OR ARCHIVE? ? OR (INFORMATION OR DAT-
             A)()MANAG??? OR DATE()(MART? ? OR WAREHOUSE? ?) OR DATAMART? ?
              OR CLUSTER???)
S14        53   S1(50N)S8(50N)(DATABASE? ? OR DATA()BASE? ? OR DBMS OR RDB-
             MS OR REPOSITOR??? OR LIBRAR??? OR ARCHIVE? ? OR (INFORMATION
             OR DATA)()MANAG??? OR DATE()(MART? ? OR WAREHOUSE? ?) OR DATA-
             MART? ? OR CLUSTER???)
S15        33   RD (unique items)
S16        21   S15 NOT (S7 OR PY=1999:2004)
```

7/3,K/1 (Item 1 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01915944 SUPPLIER NUMBER: 18129663 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Apple ships Unix Network Servers. (Apple Network Server 700)(Product Announcement)(Brief Article)
MacWEEK, v10, n12, p4(1)
March 25, 1996
DOCUMENT TYPE: Product Announcement Brief Article ISSN: 0892-8118
LANGUAGE: English RECORD TYPE: Fulltext
WORD COUNT: 146 LINE COUNT: 00013

The Unix **boxes** are available in **several versions** . An entry-level, diskless Network Server 500 costs $8,819 and includes a 132-MHz PowerPC 604 processor with 32 Mbytes of RAM, 512 Kbytes of Level 2 **cache** and a quadruple-speed CD-ROM mechanism.
A base Network Server 700 goes for $11,829 and...

7/3,K/2 (Item 2 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01801478 SUPPLIER NUMBER: 17098620 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Tatung's 20A/61 SuperCompstation.(SPARCstation-compatible system)(Evaluation)
Barker, Ralph
UNIX Review, v13, n5, p47(6)
May, 1995
DOCUMENT TYPE: Evaluation ISSN: 0742-3136 LANGUAGE: English
RECORD TYPE: Fulltext; Abstract
WORD COUNT: 3137 LINE COUNT: 00260

...ABSTRACT: SPARCstation-compatible system that offers good performance but is somewhat overpriced. It comes with one Superscalar SPARC **Version** 6 CPU and can be upgraded to **multiple** processors. The case is a 'pizza **box** ' design. There is a total of 36KB of on-chip **cache** and 1MB of secondary external **cache** . Experienced system administrators will find installation easy. Tatung pre-installs the buyer's choice of system software...

7/3,K/3 (Item 3 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01694666 SUPPLIER NUMBER: 16178464 (USE FORMAT 7 OR 9 FOR FULL TEXT)
FreeHand 4.0 for Windows: attack of the killer bezagons. (Aldus Corp's draw software) (Software Review) (New! Software)(Evaluation) (Brief Article)
Lake, Matthew J.
PC-Computing, v7, n9, p83(1)
Sept, 1994
DOCUMENT TYPE: Brief Article ISSN: 0899-1847 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT
WORD COUNT: 534 LINE COUNT: 00040

... 800) 274-7243, (206) 622-5500
Overwhelmed by the bulk of CorelDraw? Turned off by the elitist **cachet** of Adobe's Illustrator? Find Aldus FreeHand far too complicated?
Relief is in sight: Aldus FreeHand 4.0 for Windows struts its technical stuff without forcing you through the byzantine procedures of **previous versions** of the program. Gone are dozens of the dialog **boxes** you once had to navigate to perform common tasks. In their place are floating tool palettes and...

7/3,K/4 (Item 4 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01527350 SUPPLIER NUMBER: 12485907 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Bases high. (Borland International Inc.'s Paradox 4.0 database management system) (Software Review) (DOS Databases) (Evaluation)
Evans, Phil
PC User, n188, p43(2)
July 1, 1992
DOCUMENT TYPE: Evaluation ISSN: 0263-5720 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 1423 LINE COUNT: 00118

... 4.0 can be run from OS/2 2.0, either full-screen or in a DOS **box** . It won't run with **earlier** **versions** of OS/2.
The Vcache disk **cache** is incompatible with Paradox 4.0 and shouldn't be used. Also, the EMM386.EXE as supplied...

7/3,K/5 (Item 5 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01427026 SUPPLIER NUMBER: 10580966 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Buffered printer-sharing devices. (building workgroup solutions) (Hardware Review) (overview of nine evaluations of printer-sharing products)(includes related articles on Editor's Choice, how testing was done) (evaluation)
Stone, M. David
PC Magazine, v10, n8, p283(22)
April 30, 1991
DOCUMENT TYPE: evaluation ISSN: 0888-8507 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 3385 LINE COUNT: 00253

... our testing were excluded because they were OEM versions of Excellink's product, PrintPoint 6x2. Similarly, Black **Box** 's MicroSwitch 9 Plus is an OEM **version** of an **earlier** Rose Electronics model.
WHAT'S HERE
Buffered printer-sharing devices, or more simply, printer sharers, are distinguished from simple mechanical and electronic switches in...

7/3,K/6 (Item 6 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01358994 SUPPLIER NUMBER: 08182294 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Best buy: 286 personal computers. (Hardware Review) (overview of three evaluations of microcomputers)(includes related article 'Best buy user: Wendover Underwriting Agency Limited') (evaluation)
Jarrett, Dennis
Which Computer?, p50(7)
Feb, 1990
DOCUMENT TYPE: evaluation ISSN: 0140-3435 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 2274 LINE COUNT: 00172

... s a year's on-site warranty, and the manuals are really good.
* The Multiplex 286-12 **Cache** takes a sligthly different approach. It's a strongly built, tallish baby AT **box** with Digital Research's **version** of DOS, a megabyte of memory, **two** serial and two printer ports, and a grim parcel of documentation that we're told is being...

7/3,K/7 (Item 7 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)

01242841 SUPPLIER NUMBER: 06537959 (USE FORMAT 7 OR 9 FOR FULL TEXT)
EMS support improves Microsoft Windows 2.0 application performance. (Expanded Memory Specification)
Yao, Paul
Microsoft Systems Journal, v3, n1, p57(10)
Jan, 1988
ISSN: 0889-9932 LANGUAGE: ENGLISH RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 5021 LINE COUNT: 00393

... purpose. Code, resource, and other discardable segments can be destroyed and quickly re-read from the disk **cache** using the loading and discarding capabilities of the memory manager. Windows 2.0 includes a disk **cacher** , which dynamically allocates and frees EMS pages as needed. Unlike a RAM drive, optimum performance of the disk **cache** does not depend on a user decision.

Windows 2.0, unlike the **earlier versions** , will run in the compatibility **box** of OS/2. Even in this mode, Windows will use expanded memory.

How EMS Support Works

The...

7/3,K/8 (Item 1 from file: 621)
DIALOG(R)File 621:Gale Group New Prod.Annou.(R)

03799570 Supplier Number: 122251058 (USE FORMAT 7 FOR FULLTEXT)
JBoss, Inc. Delivers JBoss Application Server 4.0 to the Enterprise Market.
Business Wire, pNA
Sept 20, 2004
Language: English Record Type: Fulltext
Document Type: Newswire; Trade
Word Count: 929

... open source products, all backed with support from JBoss, Inc.:

-- Tomcat 5, the leading open source Web **container** . The integrated **version** offers **additional** value to the standalone product such as high performance, transactional HttpSession clustering over JBossCache.

-- JBossCache, the first open source transactional, distributed **cache** for fine-grained Java objects.

-- Hibernate 2.1, the popular ORM engine, which enables POJO persistence and...

7/3,K/9 (Item 1 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

09203305 Supplier Number: 78175825 (USE FORMAT 7 FOR FULLTEXT)
Getting to grips with Net security.
Pearson, Stuart
Internet Magazine, p97
March, 2001
Language: English Record Type: Fulltext
Document Type: Magazine/Journal; General Trade
Word Count: 2550

... you do. There are history files for program and document use, temporary files, swap files, backups and **previous** file **versions** , **caches** , 'deleted' email in trash files, the recycle **bin** , and a whole lot more. Then there are files others put on your PC such as cookies...

7/3,K/10 (Item 2 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

07207664 Supplier Number: 61433415 (USE FORMAT 7 FOR FULLTEXT)
Recipe for a cache bake-off; Take 20 products, turn up the heat, cook for 14 hours.(Hardware Review)(Evaluation)
Rousskov, Duane Wessels And Glenn Chisholm Alex
Network World, p60
April 10, 2000
Language: English Record Type: Fulltext
Article Type: Evaluation
Document Type: Tabloid; Trade
Word Count: 2703

... a box may be able to achieve 100 requests per second throughput with a 50% hit ratio. **Cache** misses force delays, so if a product's hit ratio goes down, its response time goes up. But that same **box** could probably get 125 requests per **second** with a 35% hit ratio.

Squid **Version** 2.4 and Swell's Tsunami CPX-1000 - a Squid-based product - scored perfect document hit ratios. Cisco's **Cache** Engine 590 was a close second at 54.8% and Cisco's **Cache** Engine 7300 was third at 54.5%. Also scoring above 52% were Pionex, iMimic, Compaq's TaskSmart...

7/3,K/11 (Item 3 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

04777385 Supplier Number: 47032518 (USE FORMAT 7 FOR FULLTEXT)
Set-top-box design needs reassessment
Lowe, Richard P. F.
Electronic Engineering Times, p68
Jan 13, 1997
Language: English Record Type: Fulltext
Document Type: Magazine/Journal; Trade
Word Count: 1657

... Hitachi (SH) family of RISC processors could handle the data rates being transmitted to a set-top **box** . There were **several** specific concerns:

- Would a low-cost **version** of the SH processor be powerful enough to handle the throughput rate coming from the demux?
- For cost considerations, what is the minimum size of the ring **buffer** that would be sufficient to handle the incoming data?
- What would be the impact of user input...

7/3,K/12 (Item 4 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)

01534835 Supplier Number: 41872749 (USE FORMAT 7 FOR FULLTEXT)
Sierra samples chip for caller-ID services
Electronic Engineering Times, p14
Feb 18, 1991
Language: English Record Type: Fulltext
Document Type: Magazine/Journal; Trade
Word Count: 857

... to be designed into standard and cordless phones, as well as into the stand-alone caller-ID **boxes** some manufacturers have sampled. Sierra will sell the chip in **several** **different** **versions** at about $2 each in high volumes.

Sierra used its cell-based design tools to take a...

...shift key demodulator from a standard modem, and combine it with a four-pole bandpass filter, input **buffer** , energy-detection circuit and

clock generator.
Companies such as Exar Corp. (San Jose, Calif.), Dallas Semiconductor Corp...

7/3,K/13 (Item 1 from file: 148)
DIALOG(R)File 148:Gale Group Trade & Industry DB

09830934 SUPPLIER NUMBER: 17785645 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Remote software offered in a 32-bit version. (Symantec Corp.'s new Norton pcAnywhere32 remote access software)
Douglass, Michelle
Computer Dealer News, v11, n24, p10(1)
Nov 29, 1995
ISSN: 1184-2369 LANGUAGE: English RECORD TYPE: Fulltext
WORD COUNT: 512 LINE COUNT: 00044

... screen; ColorScale, so the user may select higher screen refresh speeds while reducing bitmap color resolution; persistent **caching** , so users may reduce the data sent; Speed-Send, to transfer only the parts of the file that have changed; and, the ability to disable host wallpaper, screen savers and full windows dragging.
This **version** also supports file transfer between **two** computers via a parallel cable, which is included in the **box** .
For the first time in its remote control software, Symantec includes built-in Norton anti-virus scanning...

7/3,K/14 (Item 1 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)

02375387 126532331
Issues in transaction-time temporal object database systems
Norvag, Kjetil
Journal of Database Management v12n4 PP: 40-51 Oct-Dec 2001
ISSN: 1063-8016 JRNL CODE: DAN
WORD COUNT: 8155

...TEXT: where time is used as the key. ODs with different OIDs can be hashed to the same **bucket** , and for each **OlD** we have a **separate version** tree. The **version** trees are chained in a linear list. With an appropriate size of the hash table, the number of OIDs hashed to the same bucket should be low.

The architecture of the OD **cache** lookup index is illustrated in Figure 3. Each entry in the hash table is a pointer to...

7/3,K/15 (Item 2 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)

02010903 52522813
Recipe for a cache bake-off
Rousskov, Alex; Wessels, Duane; Chisholm, Glenn
Network World v17n15 PP: 62-66 Apr 10, 2000
ISSN: 0887-7661 JRNL CODE: NWW
WORD COUNT: 3190

...TEXT: a box may be able to achieve 100 requests per second throughput with a 50% hit ratio. **Cache** misses force delays, so if a product's hit ratio goes down, its response time goes up. But that same **box** could probably get 125 requests per **second** with a 35% hit ratio.

Squid **Version** 2.4 and Swell's Tsunami CPX1000 - a Squid-based product - scored perfect document hit ratios. Cisco's **Cache** Engine 590 was a close

...shift key demodulator from a standard modem, and combine it with a four-pole bandpass filter, input **buffer** , energy-detection circuit and clock generator.

Sierra has two versions of the caller-ID chip: a 14...

7/3,K/19 (Item 3 from file: 647)
DIALOG(R)File 647:CMP Computer Fulltext

00588747 CMP ACCESSION NUMBER: EET19910218S1634
Sierra samples chip for caller-ID services
LORING WIRBEL
ELECTRONIC ENGINEERING TIMES, 1991, n 629, 14
PUBLICATION DATE: 910218
JOURNAL CODE: EET LANGUAGE: English
RECORD TYPE: Fulltext
SECTION HEADING: News
WORD COUNT: 878

... packaging, to be designed into standard and cordless phones, as well as into the standalone caller-ID **boxes** some manufacturers have sampled. Sierra will sell the chip in **several different versions** at about $2 each in high volumes.

Sierra used its cell-based design tools to take a...

...shift key demodulator from a standard modem, and combine it with a four-pole bandpass filter, input **buffer** , energy-detection circuit and clock generator.

Companies such as Exar Corp. (San Jose, Calif.), Dallas Semiconductor Corp...

7/3,K/20 (Item 1 from file: 674)
DIALOG(R)File 674:Computer News Fulltext

063240
Briefs
Briefs
Journal: Computerworld Page Number: 77
Publication Date: November 17, 1997
Word Count: 259 Line Count: 24

Text:
... to achieve high-speed movement of data between mainframes and Unix or Windows NT servers. The new **version** supports four SCSI server connections and **two** mainframe connections in one **box** . Pricing ranges from $12,000 to $35,000.

NECexpands Versa line

NEC Technologies, Inc. in Mountain...

... series. The notebook has a 233-MHz Pentium processor with MMX technology; 512K bytes of Level 2 **cache** ; a hard drive that can be scaled up to 4G bytes; a 12.1-in. active-matrix...

7/3,K/21 (Item 1 from file: 810)
DIALOG(R)File 810:Business Wire

0124199 BW625

IOMEGA CORP 2: Iomega announces new pricing and new product marketing program for the Macintosh market

April 10, 1989

Byline: Business Editors/Computer Writers

...exceed any other removable products on the market. We have enhanced our proven ruggedness and reliability with **caching** , backup and utilities that will set a new level of expectation in removable storage for the Macintosh...

...vice
president of product marketing.

He further said: "As a high performance, entry level solution our Bernoulli **Box** II (20 megabyte **version**) has **many** of the performance enhancements of the 44. Its new pricing is as much as 35 percent less...

second at 54.8% and Cisco's **Cache** Engine 7300 was third at 54.5%.Also scoring above 52% were Pionex, iMimic, Compaq's TaskSmart...

7/3,K/16 (Item 3 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)

00921479 95-70871
HP extends 3000, drops software prices by allowing smaller user-license agreements
Halper, Mark
Computerworld v28n41 PP: 75 Oct 10, 1994
ISSN: 0010-4841 JRNL CODE: COW
WORD COUNT: 532

...TEXT: to 1,000 users simultaneously--100 more than the 9x8 line's previous limit. HP quadrupled memory **cache** from 256K bytes to 1M byte. An LX **version** includes **two** I/O slots, and an RX **version** has four I/O slots.

Pricing for the new **box** starts at $90,000 for a 64-user license, a 1G-byte hard drive, a 2G-byte...

7/3,K/17 (Item 1 from file: 647)
DIALOG(R)File 647:CMP Computer Fulltext

01116081 CMP ACCESSION NUMBER: EET19970113S0085
Set-top-box design needs reassessment
Richard P. F. Lowe , Senior Systems Engineer, Eytan Carmiel, Director of Software Development, Cardtools Systems Corp., San Jose, Calif.
ELECTRONIC ENGINEERING TIMES, 1997, n 936, PG68
PUBLICATION DATE: 970113
JOURNAL CODE: EET LANGUAGE: English
RECORD TYPE: Fulltext
SECTION HEADING: Embedded Systems
WORD COUNT: 1652

... Hitachi (SH) family of RISC processors could handle the data rates being transmitted to a set-top **box** . There were **several** specific concerns:
- Would a low-cost **version** of the SH processor be powerful enough to handle the throughput rate coming from the demux?
- For cost considerations, what is the minimum size of the ring **buffer** that would be sufficient to handle the incoming data?
- What would be the impact of user input...

7/3,K/18 (Item 2 from file: 647)
DIALOG(R)File 647:CMP Computer Fulltext

00589789 CMP ACCESSION NUMBER: EWN19910304S2687
This analog phone chip supports caller identification services
ELECTRONIC WORLD NEWS, 1991, n 035, 30
PUBLICATION DATE: 910304
JOURNAL CODE: EWN LANGUAGE: English
RECORD TYPE: Fulltext
SECTION HEADING: Products
WORD COUNT: 252

... to be designed into standard and cordless phones, as well as into the stand-alone caller-ID **boxes** some manufacturers have sampled.
Sierra will sell the chip in **several different versions** at around $2 each in high volumes.
Sierra used its cell-based design tools to take a...